Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática,
2008

Sérgio Tafula

Sistema de informação para apoio à arbitragem no futebol robótico

Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática, **2008**

Sérgio Miguel das Neves Tafula

Sistema de informação para apoio à arbitragem no futebol robótico

M.I.E.E.T

Dissertação apresentada à Universidade de Aveiro, em Julho de 2008, para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações (M.I.E.E.T.), realizada sob a orientação científica do Professor Artur Pereira e do Professor José Luís Azevedo, Professores Auxiliares do Departamento de Electrónica Telecomunicações e Informática da Universidade de Aveiro, e com a colaboração do Eng. Gustavo Corrente.

o júri / the jury

presidente / president

António Rui de Oliveira e Silva Borges
Professor Associado da Universidade de Aveiro

vogais / examiners committee

Artur José Carneiro Pereira
Professor Auxiliar da Universidade de Aveiro

José Luis Costa Pinto de Azevedo
Professor Auxiliar da Universidade de Aveiro

Armando Jorge Miranda de Sousa
Professor Auxiliar da Faculdade de Engenharia da Universidade do Porto

Resumo

Esta dissertação descreve e contextualiza o desenvolvimento de uma proposta para um novo sistema de apoio à arbitragem no futebol robótico, denominado de *Referee Box*. Para o efeito, foi estudado o universo de aplicação e as actuais versões do sistema. Em consequência com este estudo foi desenhado, construído e testado um sistema mais funcional e ergonómico, cuja singularidade reside na sua arquitectura modular, concorrente e distribuída.

Abstract

This thesis describes the background and the development of a proposal for a new system to support the refereeing process in the robotic soccer environment, known as *Referee Box*. To achieve that purpose, the universe of application was studied as well the current versions of a similar system. As a result of this study a system more functional and ergonomic was designed, implemented and tested. The system uniqueness lies in its modular, concurrent and distributed architecture.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Futebol robótico

O futebol emociona e preenche o imaginário de milhões de pessoas por todo o mundo. O futebol robótico é uma área da robótica que capitaliza esses fenómenos para atrair a atenção do público ao mesmo tempo que contribui para a inovação tecnológica e para a sociedade do conhecimento.

Hoje, o futebol robótico é um misto de desporto tecnológico e projecto de investigação, numa área particularmente transversal como a robótica. Nos desportos como o futebol robótico, automobilismo e motociclismo, entre outros, a tecnologia é integrada e desenvolvida em sistemas desenhados para competir entre si. Demonstra-se assim, qual o sistema que tem o melhor suporte humano e integra a melhor técnica.

Por regra o futebol robótico visa demonstrar o estado da arte e proporcionar um ambiente favorável à investigação científica, contudo os factores espectáculo e competição estão muito presentes. A tecnologia de ponta integrada na área, bem como a espectacularidade dos jogos, adicionadas às potenciais emoções e clubismos do futebol, tornam o futebol robótico uma ponte privilegiada entre o mundo da investigação científica e a sociedade. No futuro, um pouco à imagem dos desportos motorizados, em que a máquina e o homem que a cria são os protagonistas, o futebol robótico poderá ser uma montra de tecnologia e um alvo apetecido da curiosidade das massas.

1.1.1 RoboCup, um desafio.

O RoboCup [22] é uma iniciativa internacional de educação e investigação. Nasceu de um projecto internacional conjunto fundado em 1993. Actualmente, é composto maioritariamente pela comunidade académica. Pretende promover o desenvolvimento da inteligência artificial e áreas relacionadas, bem como integrar as mais variadas tecnologias num sistema que tenta resolver um desafio comum.

O principal desafio comum que o RoboCup propõe é o futebol robótico. Com ele pretende-se desenvolver e mostrar inovações destinadas a serem aplicadas na indústria, noutros projectos académicos e em aplicações socialmente relevantes. À data, as universidades e indústrias de base tecnológica são os principais patrocinadores do RoboCup

e dos eventos com ele relacionados.

O RoboCup estabeleceu como principal meta para o futebol robótico, ter em 2050 uma equipa de robôs humanóides que se bata de igual para igual com a equipa campeã do mundo de futebol.

Não esquecendo o seu objectivo final, o RoboCup tem de se adaptar à tecnologia, orçamento e meios humanos existentes. Mesmo tendo em conta que hoje a alta integração de poder computacional em dispositivos cada vez mais pequenos, leves, eficientes e baratos torna viável construir equipas de robôs capazes de jogar um jogo muito parecido com o futebol, continua a ser necessário simplificar os sistemas e fazer opções. Algumas equipas focam o desempenho dos robôs na táctica e eficiência de jogo (robôs com rodas e simulação), outras tentam uma maior aproximação ao jogadores do futebol convencional (robôs humanóides).

Em teoria existe tecnologia para construir equipas de futebol com robôs humanóides que realmente se assemelham a equipas convencionais, mas mesmo com o mais alto investimento concebível, no actual estado da arte da robótica, o desempenho não se aproximaria de uma equipa humana. Um projecto de engenharia tem de ser razoável e ter alguma mais valia que o justifique. A relação custo/eficiência de uma equipa destas é hoje demasiado alta para justificar a sua construção.

Apesar de existirem progressos notáveis na robótica humanóide não associada ao futebol robótico, onde se destacam os robôs bípedes como o ASIMO [1] capaz de correr, subir escadas, curvar, compensar desequilíbrios, etc... a tecnologia actual não permite construir jogadores de futebol robótico com forma e desempenho semelhantes à de um jogador humano. Estamos muito longe que isso aconteça e a "indústria" que sustenta o futebol robótico não tem capacidade para que isso venha a acontecer num futuro próximo. Esta afirmação é sustentada também pelo facto de que pôr robôs a jogar futebol não passa exclusivamente por desenvolver um robô humanóide com agilidade e velocidade de um ser humano. Este jogo é colectivo, tem uma componente táctica e cooperativa muito importante. Para que o futebol robótico se aproxime do futebol convencional é necessário que os robôs futebolistas aperfeiçoem muitas dessas valências.

Resultado de alguns condicionalismos económicos, os robôs futebolistas, em regra, não têm *hardware* de ponta, mas não estão muito longe disso. A filosofia da maioria das equipas consiste em usar *hardware* comercial (usado na indústria e na electrónica de consumo) integrado em *softwares* pioneiros e em configurações inovadoras. Hoje o acréscimo de valor criado no seio do futebol robótico reside mais no *software* e na arquitectura dos sistemas do que no desenvolvimento de novos componentes.

O desafio de futebol robótico do RoboCup e as suas regras, são a referência de todas as restantes competições de futebol robótico, essencialmente porque é neste desafio que se concentra a tecnologia de ponta e a elite do meio, pelo que as restantes competições se definem à imagem desta. Com as devidas distâncias, o RoboCup está para o futebol robótico da mesma maneira que a FIFA [1] está para o futebol convencional. À semelhança da FIFA, o RoboCup organiza a competição de referência e suporta um comité que dita as regras.

Tendo em conta a tecnologia, recursos existentes e os objectivos traçados, o Robo-

[1] F.I.F.A. - Fédération Internationale de Football Association [9]

Cup dividiu as competições do futebol robótico em várias ligas. Isto serve sobretudo para atenuar a transversalidade dos sistemas e permitir que diferentes equipas concentrem os seus esforços em diferentes áreas. Quando os sistemas tácticos e a robótica humanóide puderem conviver facilmente no mesmo sistema, pretende-se fundir os diferentes ramos.

O traço comum de todas as ligas de futebol robótico do RoboCup é a autonomia dos sistemas, pelo que é proibida qualquer intervenção humana depois do jogo começar. A excepção são os eventos de arbitragem e a colocação e retirada dos robôs em campo.

Das ligas do RoboCup destaca-se a M.S.L.(*Middle Size League*) pela sua actual competitividade e dimensão.

Segue-se uma breve descrição das várias ligas do RoboCup.

Liga de simulação

Nesta liga [14] joga-se num ambiente virtual ou simulado num programa de computador. A liga de simulação tem como principal objectivo desenvolver os algoritmos de jogo relacionados com a estratégia e comportamentos cooperantes, libertando as equipas da complexidade da aquisição real de dados e da movimentação de um robô.

A liga de simulação fornece hoje aos participantes um ambiente muito completo e realista para a construção de algoritmos evoluídos, mais concretamente, fornece um motor físico e exibe um ambiente 3D que permite simular um humanóide com quase todas as articulações. Cabe ao programador controlar todas estas articulações, assim como os inúmeros comportamentos, de maneira a construir jogadores e equipas virtuais muito completas.

(a) Jogo da liga de simulação

(b) Exemplo de um visualizador 3D para os jogos da liga de simulação

Figura 1.1: Liga de Simulação.

Liga dos robôs pequenos

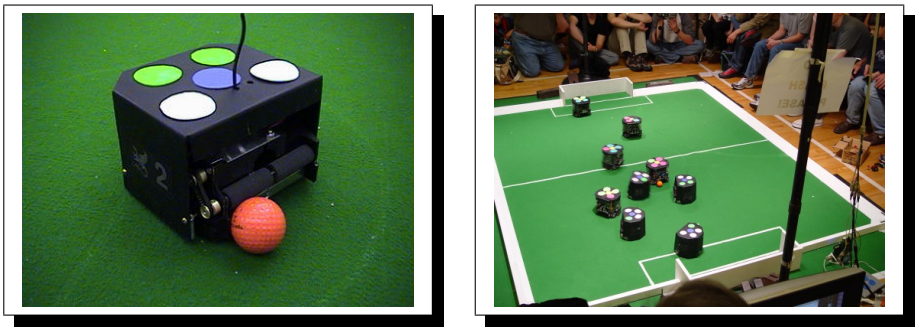
A Liga dos robôs pequenos (Small Size League) [17] é a liga para os robôs com, no máximo, 180mm de diâmetro e 150mm de altura, podendo esta altura ser ultrapassada se existir visão integrada no robô.

Os robôs jogam num campo verde alcatifado com 6,1m de comprimento e 4,2m de largura, com uma bola laranja, várias marcas nos robôs e no campo. Há equipas que usam um sistema de visão global que está sobre o campo e outras que usam visão local sobre os robôs, contudo a primeira é muito mais comum.

A informação recolhida no robô pode ser processada dentro deste ou ser transmitida para processamento num computador situado fora do campo, denominado por estação base. Este computador é responsável por recolher os dados de arbitragem e a imagem da câmara global, se esta for usada.

Na maior parte dos casos a estação base é responsável pela quase totalidade do processamento e posicionamento dos pequenos robôs no campo, enviando esta informação para os robôs via comandos de rádio [5].

Nesta liga recorre-se a uma configuração de *hardware* simplista. Usa-se o maior número possível de componentes comerciais, como os computadores, câmaras e controlos rádio. Pretende-se materializar a competição, mas evitar muitos problemas de miniaturização, aquisição de dados e autonomia dos sistemas. Esta abordagem permite que as equipas que adiram a esta liga possam concentrar-se sobretudo no *software* e em questões de algoritmia.



(a) Robô da liga dos robôs pequenos (b) Jogo da liga dos robôs pequenos

Figura 1.2: Liga dos robôs Pequenos - *Small Size League (S.S.L.)*.

Liga dos robôs médios

A Liga dos robôs médios (Medium Size League) [16] é a liga para os robôs que caibam num quadrado de 30cm a 50cm de lado, 40cm a 80cm de altura e um máximo de 40Kg de peso.

Os robôs jogam num campo verde alcatifado com 8m a 16m de comprimento e 6m a 12m de largura, com uma bola cor-de-laranja. A cada equipa é atribuída uma cor. A distinção dos robôs em campo é feita através da colocação de faixas coloridas na parte superior dos robôs. Cada equipa é composta por 4 a 6 robôs de campo e 3 suplentes [18].

Nesta liga também existe, fora do campo, uma estação base para cada equipa, mas a sua exclusiva tarefa é monitorizar e coordenar a equipa ao nível do treinador, bem como retransmitir para os robôs a informação proveniente da equipa de arbitragem.

Na liga dos robôs médios toda a captação de dados, predominantemente visuais, e todo o processamento são feitos a bordo dos robôs.

Figura 1.3: Liga dos Robôs Médios (Robótica 2008 - Aveiro).

O sistema de comunicação entre robôs, e entre estações base e robôs é feito via rede *wireless*, mas entre a *Referee Box* e estações base é feita via rede de cabo.

A liga dos robôs médios integra todas as áreas necessárias para construir um máquina completamente autónoma para jogar futebol. Nesta liga recorre-se a robôs com rodas e a mecanismos para manipular a bola, que simplificam o controlo dos robôs, diminuem o realismo, mas deixam mais espaço para tácticas e processamento de dados de jogo.

Liga dos robôs com quatro pernas / Liga de plataforma comum

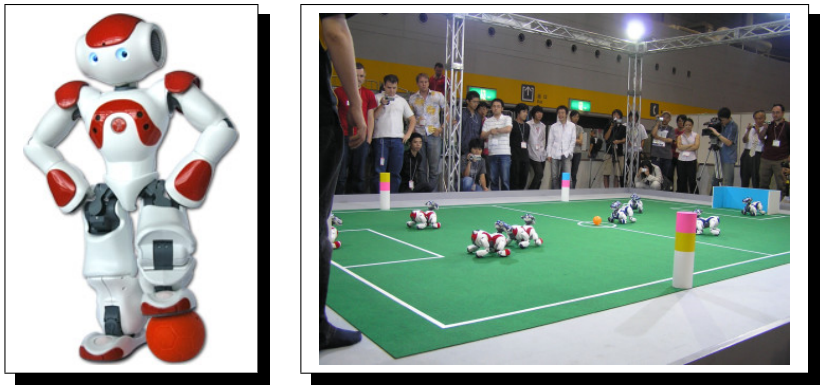
A liga dos robôs com quatro pernas consiste em usar vários Sony Aibo a jogar em equipa num campo de 6,9m por 4,6m com características semelhantes ao das ligas anteriores [6].

Esta liga apresenta-se como uma via para integrar este *hardware* de muito fácil acesso, permitindo que equipas entrem no desafio e em competição com robôs, mas sem nenhum desenvolvimento de *hardware*.

Esta liga apresenta-se como um desafio consideravelmente diferente, na medida em que controlar um quadrúpede num jogo de futebol tem várias especificidades na coordenação de movimentos, nomeadamente no chuto e na deslocação. Os robôs da Sony fornecem algumas ferramentas para a sua programação, contudo têm limitações no que toca ao desempenho porque foram claramente desenhados para ser um brinquedo, não um jogador de futebol.

A liga dos robôs de quatro pernas será substituída pela liga de plataforma comum[7]. O descontinuado robô de quatro pernas da Sony será substituído pelo robô humanóide da Aldebaran. Todavia as principais características da liga mantêm-se, ou seja a

ausência de desenvolvimento de *hardware*, a autonomia, a visão e as comunicações [7].



(a) Robô Aldebaran (b) Jogo da liga dos robôs de quatro pernas

Figura 1.4: Liga dos robôs com quatro pernas / Liga dos robôs com de plataforma comum.

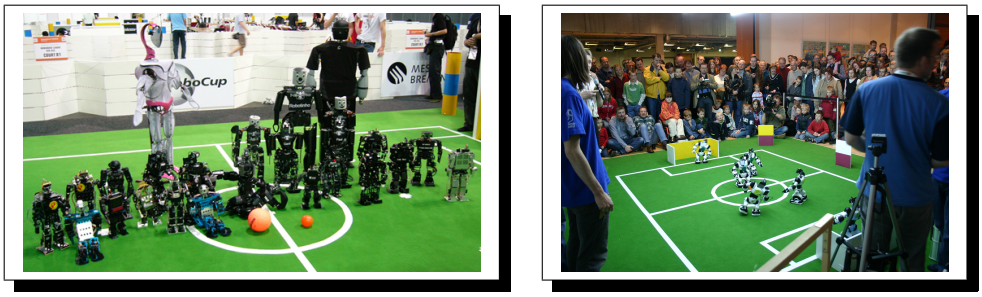
Liga dos robôs humanóides

A liga dos robôs humanóides [15] pretende ser o corolário de todas as outras, dado que o desafio proposto pelo RoboCup é construir um robô o mais parecido possível com um humano e uma equipa com as melhores características possíveis.

Esta liga é dividida em dois tamanhos: o tamanho de criança, onde os robôs têm 300 a 600mm de altura e o tamanho de adolescente onde os robôs têm 659 a 1300mm de altura. Existem também competições para vários gestos técnicos, entre eles o penálti. Os robôs jogam num campo de 450cm a 600cm de comprimento e 300cm a 400cm de largura com características semelhantes ao das anteriores ligas [23].

Nesta liga são bem patentes as limitações da tecnologia actual: gestos como ver, andar e correr, que são banais para a esmagadora maioria dos humanos, envolvem uma complexidade enorme para um robô. Conjugar todas estas complexidades com a resultante de articular uma equipa em jogo, é um desafio de concretização verdadeiramente ambicioso.

Na liga humanóide são usados muitos robôs comerciais, não desenhados de raiz para jogar futebol. Esta solução apresenta-se como uma via para ampliar a competição, libertando as equipas do muito complexo desenvolvimento do *hardware* humanóide e do respectivo controlo de baixo nível. Todavia, quando se usam soluções comerciais normalmente abdica-se de algum desempenho. Estes robôs destinam-se ao mercado infantil ou pedagógico, onde é suposto haver razoabilidade económica e facilidade de uso, o que tem como consequência uma disponibilidade limitada de recursos.



(a) Robôs da liga humanóide (b) Jogo da liga humanóide.

Figura 1.5: Liga Humanóide.

1.1.2 O sistema de arbitragem do RoboCup

À semelhança de uma partida de futebol convencional, numa competição de futebol robótico participa uma equipa de arbitragem humana. Contudo, na maioria das ligas do futebol robótico, existe um sistema informático, de nome *Referee Box*, que tem como função principal servir de ponte entre humanos e robôs, transmitindo às equipas os eventos de arbitragem produzidos pelos árbitros. Este sistema tem a função acessória de guardar e catalogar todas as informações do jogo, para uso futuro.

No modelo actual da liga dos robôs médios, a equipa de arbitragem é constituída normalmente por um árbitro de campo, opcionalmente um ou dois árbitros auxiliares e por um operador (*time keeper*) da *Referee Box*. Tradicionalmente um dos auxiliares controla se a bola sai de campo (*line watcher*) e o outro controla os elementos humanos das equipas, ou seja verifica se estes durante a partida comandam os robôs (*keyboard watcher*). Nesta modalidade o árbitro principal é o observador principal e juiz do jogo (figura 1.6(a)), que apoia as suas decisões nos restantes três auxiliares. O árbitro principal tem a responsabilidade de recolher dados, ordenar e transmitir todas as suas instruções ao operador da *Referee Box*, que tem de as introduzir em tempo útil no sistema.

Hoje a *Referee Box* da liga dos robôs médios é responsável por armazenar e processar os dados inseridos, assim como encaminhá-los para as estações base das equipas. Estas, por sua vez, são responsáveis por retransmitir essas instruções para os seus robôs. O sistema de arbitragem da M.S.L. do RoboCup está resumido no esquema da figura 1.6(b).

O público e o operador da *Referee Box* ouvem as indicações verbais e os apitos e vêem a sinalética do árbitro principal para se integrarem do jogo. Actualmente o operador da *Referee Box* além de introduzir as ordens no sistema, tem também a responsabilidade de actualizar o painel onde é exibido ao público o resultado do jogo.

1.1.3 Eventos de arbitragem do RoboCup

No futebol o árbitro é o garante do cumprimento das regras do jogo. As regras do futebol robótico são aplicadas por humanos e derivam das regras oficiais do futebol convencional. Partindo do regulamento oficial da FIFA o RoboCup cria um regulamento

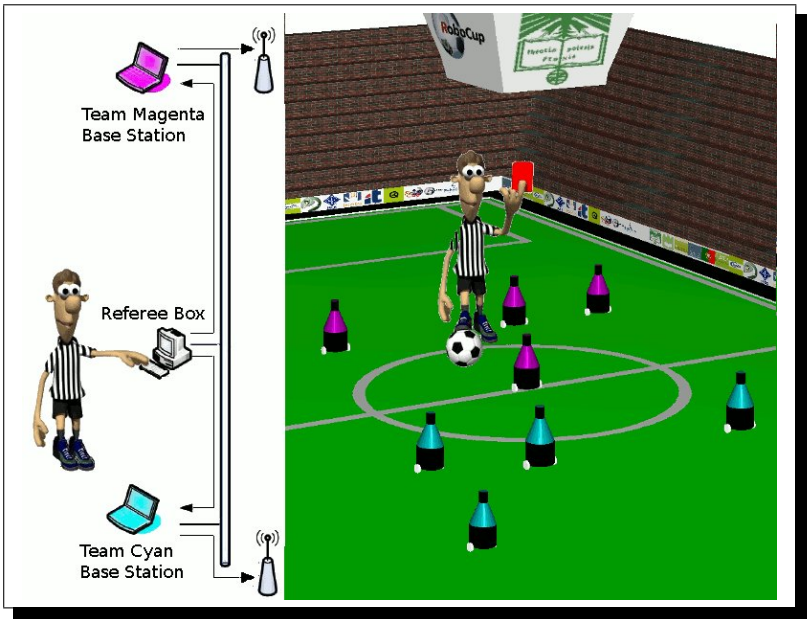
para cada uma das suas ligas. A filosofia para todas as ligas, e consequentemente para todos os regulamentos, é produzir com a tecnologia e intervenientes actuais um jogo o mais parecido possível com o futebol convencional.

O futebol robótico pretende ser uma cópia do futebol convencional. Isto implica que a esmagadora maioria da regras do futebol robótico são familiares aos milhões de pessoas que dominam os princípios elementares do jogo de futebol convencional[9]. Daqui em diante considera-se que as regras básicas do futebol robótico [18] [5] são do conhecimento do leitor.

Só as regras que definem o fluxo de jogo, definem quais são os dados importantes deste ou geram necessidade de comunicação, são relevantes para a definição de requisitos da *Referee Box*.

As regras do futebol robótico quando aplicadas geram eventos de arbitragem. Entende-

(a) O árbitro principal observa o jogo dentro ou perto dos limites do campo.



(b) Esquema do sistema de arbitragem. (Esquema adaptado da apresentação do artigo [8]).

Figura 1.6: Sistema de arbitragem actual da M.S.L. do RoboCup.

se por evento de arbitragem algo que crie a necessidade de comunicação entre os árbitros e os restantes intervenientes no jogo, ou crie a necessidade de armazenar informação para uso futuro. A *Referee Box* tem, ou pode vir a ter, intervenção em todos os eventos de arbitragem, pelo que estes são o cerne das especificações deste sistema. A criação de eventos e a informação que flui no sistema em resultado destes é definida por uma ou mais regras. Este facto cria uma dualidade entre evento(s) e regra(s).

Segue-se a lista de eventos onde, segundo o regulamento da M.S.L.(*Middle Size League*)[18], a *Referee Box* pode ter alguma intervenção:

Apresentação dos intervenientes
É responsabilidade da equipa de arbitragem verificar presenças, registar equipas e seus constituintes, fazer verificações técnicas[2] e agir em conformidade depois de efectuar este levantamento. A *Referee Box* serve de meio de registo de alguns destes dados.

Início e final de Jogo
O jogo tem uma hora definida para começar e um tempo de duração. As indicações de começo e final de jogo provêm da equipa de arbitragem e são transmitidas às equipas via *Referee Box*.

Início e final de parte de jogo
O jogo pode ser dividido em três partes, mediadas por intervalos. As equipas são informadas através da *Referee Box* sobre o início e o final de todas as partes e intervalos. Os nomes das partes são:

- **Primeira parte** *(First Half)*
- **Segunda parte** *(Second Half)*
  Depois de um intervalo inicia-se a segunda parte do jogo. Se o jogo não for a eliminar, acaba no final desta parte.
- **Marcação de penáltis** *(Penalty Shoot Out)*
  Se, no final da 2ª parte, o resultado for um empate e o jogo for a eliminar, depois de mais um intervalo, existe uma última parte dedicada à marcação de penáltis para desempatar o jogo.

Indicação de tempo decorrido
É tarefa da equipa de arbitragem contar o tempo e para o efeito pode socorrer-se da *Referee Box*. A *Referee Box* poderá ser usada também para enviar às equipas o tempo e período de jogo actual.

Paragem e recomeço de jogo
Sempre que o árbitro assim o entenda, um jogador ou técnico tiver de ser penalizado, houver uma substituição, reparação, cartão, golo, falta, é bola ao solo ou a meio campo, o jogo terá de ser interrompido. Todas as paragens e recomeços são transmitidos às equipas via *Referee Box*.

[2] As verificações técnicas das equipas de arbitragem do RoboCup consistem na validação das características do campo e dos robôs em jogo, nomeadamente aquelas que são visadas nos regulamentos.

Evento de jogo - (com o jogo parado)

Depois de interrompido o jogo, pode ser assinalado, via *Referee Box*, um dos seguintes eventos:

- **Golo** *(Goal)*
  É indicada a cor da equipa e o número do jogador que marcou o golo, bem como a cor da equipa que beneficiou dele.
- **Substituição** *(Substitution)*
  É indicada a cor da equipa, o número do jogador que sai e o do que entra.
- **Saída para reparação** *(Repairing Out)*
  É indicada a cor da equipa e o número do jogador que sai.
- **Entrada pós reparação** *(Repairing In)*
  É indicada a cor da equipa e o número do jogador que entra.
- **Cartão** *(Card)*
  É indicada a cor da equipa, o número do jogador e a cor do cartão.
- **Bola Fora** *(Throw In)*
  É indicada a cor da equipa que ganha a posse de bola.
- **Livre** *(Free Kick)*
  É indicada a cor da equipa que beneficia de um livre.
- **Canto** *(Corner Kick)*
  É indicada a cor da equipa que beneficia de um canto.
- **Pontapé de baliza** *(Goal Kick)*
  É indicada a cor da equipa que beneficia de um pontapé de baliza.
- **Bola no solo** *(Dropped Ball)*
  É ordenada a reposição da bola em pontos pré estabelecidos no campo.
- **Pontapé de saída** *(Kick Off)*
  É indicada a cor da equipa que tem a posse de bola e ordenado o (re)inicio do jogo a meio campo.
- **Penálti** *(Penalty)*
  É indicada a cor da equipa que tem a posse de bola e ordenada a marcação de um penálti.

Esta lista variou e presume-se que variará muito pouco, essencialmente porque as regras básicas do futebol convencional vigoram à décadas e não deverão mudar, pelo que temos um cenário estável onde as necessidades de comunicação e registo são constantes. O que pode mudar é a via como estas necessidades são satisfeitas.

À data, alguns dos itens desta lista não estão implementados nas actuais versões da *Referee Box*, nomeadamente o controlo de reparações, substituições e de cartões. Os restantes itens, apesar de estarem contemplados nas funcionalidades da *Referee Box* actual, podem ser mais amplamente cobertos, ou seja, pode passar pelo sistema mais informação sobre estes.

1.2 A *Referee Box*

O sistema informático, de nome *Referee Box*, serve de apoio à arbitragem e interliga os intervenientes num jogo de futebol robótico. Cada liga do RoboCup tem a sua *Referee Box*, por sua vez, cada uma destas *Referee Box*s tem várias versões. Este sistema, nas suas várias formas e versões, adapta-se às especificidades e regras de cada liga.

O RoboCup apresentou em 2007, um poster onde resume o passado e as previsões para o futuro da M.S.L.. Esta apresentação foi denominada de *Road Map* - RoboCup 2007 Atlanta[4].

A partir desta publicação e da história da M.S.L., conclui-se que a *Referee Box* é uma aplicação, que começou por ser usada num sistema de arbitragem, onde o árbitro comunicava **directamente** com os elementos humanos das equipas, e terminará num outro, onde o árbitro comunicará **directamente** com os robôs dessas mesmas equipas. O primeiro caso aconteceu nos primeiros jogos da M.S.L., onde alguns elementos das equipas ouviam, viam e transmitiam manualmente as ordens do árbitro aos seus robôs (através da sua estação base). O segundo caso é o cenário projectado para 2012, onde as funções de comunicação da *Referee Box* serão abolidas, sendo a comunicação efectuada com gestos e sons exibidos directamente aos robôs em campo.

Entre estas duas situações limite está a situação actual, apresentada na subsecção 1.1.2, onde a *Referee Box* é um dos elos do sistema de comunicação indirecta entre robôs e árbitros.

Em todo o caso, mesmo depois de implementada a comunicação directa entre árbitros e robôs, a *Referee Box* poderá sempre ser o sistema de apoio à arbitragem e o sistema de informação de um jogo.

As duas funcionalidades básicas da *Referee Box* actual são enumeradas de seguida.

Sistema de comunicação
: Sistema de comunicação entre humanos (árbitros) e robôs (equipas).

Sistema de Informação
: O sistema trata, cataloga e armazena de forma perene a informação introduzida, calculada e transmitida.

As funções não directamente relacionadas com a comunicação entre árbitro e robôs, são resultado do aproveitamento da infraestrutura e da informação que flui no sistema, para a criação de um registo integrado, fiável e seguro.

1.2.1 A *Referee Box* oficial

A actual *Referee Box* oficial da liga dos robôs médios do RoboCup é a *Referee Box* 2005 [19]. Esta aplicação interliga, sobre uma rede, duas equipas e um árbitro auxiliar, que é o único operador e monitor da interface gráfica deste sistema. Esta aplicação foi desenhada numa arquitectura centralizada e com uma algoritmia predominantemente em série, ou seja, a interface com as equipas, a interface com o operador e o registo de jogo estão no mesmo fio de execução, na mesma aplicação e na mesma máquina.

A introdução de informação neste sistema faz-se, quase exclusivamente, através da interface com o operador. A retirada de informação faz-se sobretudo através das mensagens enviadas às equipas (anexo B) e através de um relatório produzido no final do jogo (anexo E).

1.3 Motivação

O projecto CAMBADA[3], enquadrado na Actividade Transversal em Robótica Inteligente [2] do IEETA / Universidade de Aveiro[4], desenvolve desde 2003 uma equipa de futebol robótico, de nome igual ao do projecto, para participar na liga MSL do RoboCup. Actualmente, esta equipa compete ao mais alto nível nas competições locais de futebol robótico e no RoboCup.

(a) Robô (b) Equipa (final Robótica 2007)

Figura 1.7: CAMBADA - equipa de futebol robótico da U.A..

Os membros das equipas participantes são os melhores avaliadores das características da *Referee Box*. Por um lado, porque as estações base têm de comunicar com a *Referee Box*. Por outro lado, porque durante as competições é comum a *Referee Box* ser operada por membros das equipas. É convicção de alguns participantes que a estabilidade, a fiabilidade, o interface e a arquitectura da *Referee Box* pode ser melhorada. Foi esta constatação que motivou a criação de um projecto na U.A. para desenvolver uma nova *Referee Box*.

Perspectiva-se que a *Referee Box* num futuro próximo passará a ligar não só árbitros e equipas-robôs, mas também o público. Em suma prevê-se no futuro que a *Referee Box* se transforme num sistema de informação e de comunicação predominantemente para humanos, não para robôs como é hoje.

Para o futebol robótico ser um espectáculo mais apetecível, terá de haver mais informação para o público. Faz pouco sentido que a única informação exibida ao publico seja o resultado do jogo actualizado manualmente. Informação relativa às equipas em campo, tempos e eventos de jogo relevantes, como faltas, substituições, cartões e

[3]C.A.M.B.A.D.A. - Cooperative Autonomous Mobile roBots with Advanced Distributed Architecture [3]

[4]I.E.E.T.A. - Instituto de Engenharia Electrónica e Telemática [12] de Aveiro (Universidade de Aveiro [27])

golos, poderiam ser tratados estatisticamente e projectados em grande formato para as pessoas presentes no "estádio". A *Referee Box* é o "candidato natural" a fornecer a informação a esse sistema ou a incluir esse sistema.

Para o futebol robótico ter uma arbitragem mais rigorosa, ter-se-á que substituir o sistema de introdução manual e indirecta de dados, por um sistema mais automático e portátil. Hoje já existem sérios problemas derivados das crescentes dimensões do campo, principalmente no que toca à transferência verbal de informações entre árbitros, o que é crítico para introduzir dados no sistema. Pelo mesmo motivo e em consequência do cada vez maior número de espectadores, a percepção do que se passa pelo público e pelos elementos humanos das equipas também está a baixar. É provável que no futuro o sistema de introdução de dados seja desacoplado, criando-se para o efeito interfaces com o utilizador mais versáteis, como sistemas de som com reconhecedores de voz, comandos portáteis, apitos com frequência modulada, etc ...

Para o futebol robótico ser economicamente mais viável, poder-se-á aproveitar o provável sistema de exibição de informação para passar publicidade paga, atraindo a atenção do público para este monitor gigante com a informação estatística e de arbitragem, intervalada com animações associadas a eventos e a compromissos publicitários. A filosofia de complementar o espectáculo com informação e publicidade audiovisual é muito seguida no futebol convencional com os resultados que todos conhecemos.

A *Referee Box* 2008, após aprovação local, será publicada sob a licença G.P.L. (*General Public Licence*[11]) e proposta ao comité do RoboCup para possível adopção como sistema oficial dos jogos da liga dos robôs médios do RoboCup.

1.4 Objectivos

Pretendia-se desenvolver uma *Referee Box* para a M.S.L. tendo em consideração as alterações às regras previstas para a edição de 2008 do RoboCup.

Os objectivos eram:

- Análise do problema, identificação de requisitos e levantamento do estado da arte e do trabalho relacionado.
- Especificação da arquitectura da *Referee Box*.
- Desenvolvimento e teste da *Referee Box*.
- Construção do pacote de instalação e escrita dos manuais de instalação e utilização.
- Redacção da dissertação

1.5 Trabalho realizado

No âmbito desta dissertação foi estudada a versão oficial do sistema (*Referee Box* 2005), assim como uma proposta na linha da anterior (*Referee Box* 2007).

Definiu-se uma arquitectura distribuída e modular para o novo sistema. Definiu-se também um protocolo de comunicação baseado em XML [5] (anexo A) para as mensagens que fluem na rede que une os vários módulos.

Foram desenhados, construídos e testados os seguintes componentes da nova *Referee Box*.

- Uma biblioteca (*Referee Box Parser*) que transforma as mensagens que circulam no sistema, em dados facilmente integráveis nos restantes módulos.
- Uma aplicação servidora (*Referee Box Server*). Esta aplicação gere os recursos de rede, interliga as entidades, processa e reencaminha a informação do sistema, assim como gera toda a informação de registo.
- Uma aplicação de comando (*Referee Box Command*), que serve de interface gráfica ao operador da *Referee Box*.
- Uma aplicação de Visualização (*Referee Box Viewer*) que exibe ao público, presente no "estádio", o estado do jogo e animações.
- Uma aplicação responsável pela compatibilidade do sistema com o antigo formato (*Referee Box Proxy*).
- Uma aplicação (*Referee Box Reporter*) que transforma o *log* XML do sistema num relatório de jogo escrito em LateX. Este ficheiro TeX pode ser compilado num documento legível (PDF ou num web site HTML).

1.6 Estrutura da Dissertação

Para além da introdução, onde se introduz o leitor no universo de aplicação da *Referee Box*, o futebol robótico, mais especificamente, o RoboCup e o seu sistema de arbitragem, esta dissertação contém mais 4 capítulos e 5 apêndices.

No segundo capítulo, Sistemas Distribuídos, faz-se uma breve apresentação do conceito de distribuição de sistemas computacionais e uma comparação entre sistemas distribuídos e sistemas concentrados.

No terceiro capítulo, Arquitectura, apresenta-se a arquitectura distribuída subjacente á *Referee Box* 2008, assim como os traços gerais da sua implementação.

No quarto capítulo, Resultados, descrevem-se as aplicações desenvolvidas, nomeadamente as suas interfaces, funcionalidades e problemas.

As conclusões desta dissertação são apresentadas no último capítulo. Neste, apresenta-se também a nossa visão para o trabalho futuro, bem como as ilações que se tiraram após o uso da *Referee Box* 2008 numa competição nacional de futebol robótico, o Robótica 2008.

[5] XML *(eXtensible Markup Language)* é uma recomendação da W3C *(World Wide Web Consortium)* para gerar linguagens de marcação para necessidades especiais.

Capítulo 2

Sistemas Computacionais Distribuídos

A *Referee Box* 2008 é um dos sistemas de apoio à arbitragem, construídos no seio da liga dos robôs médios do RoboCup. A sua singularidade reside na arquitectura modular, concorrente e distribuída, motivo pelo qual, neste capítulo se faz uma análise sucinta do estado da arte dos sistemas computacionais distribuídos e dos sistemas computacionais centralizados.

2.1 Definição

A construção de sistemas computacionais distribuídos ([13], [24], [25]), assenta num paradigma arquitectural que consiste na junção de vários sistemas computacionais independentes, interligados por um canal de comunicação, que funcionam em prol de um objectivo comum. Um sistema distribuído tem três características principais:

- É constituído por múltiplos nós de processamento.
- Os nós são ligados por um canal de comunicação.
- Existe um estado partilhado entre todos os nós que define o estado do sistema.

A fronteira entre um sistema de computadores interligados e um sistema computacional distribuído não é bem definida. Para avaliar o grau de distribuição de um sistema tem-se em conta a blindagem aos seguintes eventos ou características:

- Falha de um nó de processamento.
- Falha de comunicação.
- Comunicação insegura.
- Comunicação dispendiosa.

Um sistema computacional distribuído é aquele que, apesar de aparentar ser um qualquer sistema centralizado, é constituído por vários componentes independentes e interligados. Do ponto de vista exterior (do utilizador, por exemplo), deve ocultar a sua arquitectura e deve parecer-se o mais possível com um sistema concentrado e local equivalente.

2.1.1 Definição Restrita

Existem interpretações do conceito genérico de sistema distribuído, que consideram necessárias as três características supra citatas, assim como a imunidade aos eventos apresentados em cima, para que um sistema computacional seja considerado distribuído.

Esta catalogação restrita de sistema computacional distribuído além de não se consensual, não é muito natural. Se se analisar o significado do verbo distribuir, em português e em inglês, por exemplo, apercebemo-nos que este serve para qualificar todo e qualquer acto de cisão ordenada. Na semântica deste verbo está implícito que as peças divididas pertencem a um domínio uno, mas nada se deve inferir sobre a validade deste domínio quando uma das sua peças desaparece, ou quando as suas peças não estiverem em contacto. [1] .

Existem definições de sistema distribuído que espelham este paradoxo. Um exemplo é a definição de Leslie Lamport - *"Um sistema distribuído, é aquele que pára de funcionar quando falha uma máquina da qual nunca ouvimos falar."*

No caso de se querer descrever a forma restrita do conceito, deve-se fazê-lo de forma explícita. Quando se catalogam sistemas, deve-se falar de graus de distribuição. Todavia pode-se considerar que a pureza do conceito é a catalogação restrita.

2.2 Sistemas Computacionais Distribuídos vs. Sistemas Computacionais Concentrados

Não obstante as ressalvas feitas em cima, aceita-se que os "sintomas" de um sistema distribuído se baseiam na existência de vários componentes com falhas independentes, na comunicação entre os componentes e na manutenção de um estado comum a todo o sistema.

O paradigma da distribuição sustenta-se em oposição ao da concentração. Nas primeiras quatro décadas de existência da computação, o programador e o utilizador eram muito próximos e o paradigma de computação concentrada imperava. Na década de 50 do século passado, o programador reservava durante algum tempo todos os recursos do precioso computador para si. Na década de 60, com o aparecimento das linguagens de linha de comando, o programador depositava as suas tarefas numa fila e o sistema executava-as sequencialmente. Na década de 70, os sistemas forneciam ao programador e ao utilizador uma ilusão de paralelismo, criada através da comutação

[1] Por exemplo, o facto de um bolo ter sido distribuído, não é afectado por alguém comer uma fatia, mas está implícito que as fatias foram separadas.

entre as diferentes tarefas em execução. Na década de 80, o utilizador usava sobretudo o seu computador pessoal. Na década de 90 em diante, os computadores pessoais e institucionais iniciam a sua "fusão" numa malha de proporções globais.

A evolução da tecnologia subjacente aos computadores e às comunicações, assim como a crescente disponibilidade destas, possibilitou este percurso. Quando a multiplicidade de hardware se aliou às redes de comunicações, descobriu-se que as desvantagens do modelo distribuído eram invertíveis e menos importantes do que as funcionalidades que este paradigma podia criar.

2.2.1 Desvantagens

Podem-se identificar algumas desvantagens dos sistemas computacionais distribuídos relativamente aos equivalentes concentrados. Enumera-se a seguir aquelas que o autor considera mais significativas.

Comunicação exclusivamente por mensagens

Os sistemas computacionais concentrados implementam sempre sistemas de comunicação internos, que se baseiam na transferência de informação através de um espaço de endereçamento partilhado. Nos sistemas distribuídos os componentes não partilham memória física e por regra estão afastados. Este facto faz com que a comunicação entre componentes tenha um ou vários intermediários e esteja sujeita a um conjunto de factores que afecta a sua fiabilidade. Quando o custo da comunicações entre componentes é superior ao custo de comunicação interna, a informação a transmitir deve ser condensada em mensagens e deve-se munir os sistemas de mecanismos que suportem mais o erro e a latência das comunicações.

Modelo de falhas

Dado que um sistema computacional distribuído tem um maior número de componentes do que o seu equivalente concentrado, existe um maior número de pontos de falhas. A juntar a isto, num sistema distribuído, temos de adicionar o potencial para o erro que o sistema de comunicação acrescenta.

Segurança

O maior número de componentes, em teoria, diminui a segurança física da informação e da integridade de um sistema computacional distribuído, principalmente porque há mais pontos a defender. Mais uma vez, o sistema de comunicação (normalmente grande e exposto) colabora para ampliar esta fragilidade.

Heterogeneidade

A diversidade de componentes e programas dentro de um sistema computacional distribuído, assim como a pertença destes a diferentes domínios de produção, instalação

e manutenção, é uma menos valia para um sistema distribuído, quando comparado com o seu equivalente concentrado.

Desempenho

Um sistema distribuído consome mais recursos do que o seu equivalente concentrado, uma vez que é preciso gerir a produção de mensagens, a comunicação, a compatibilidade e a segurança da informação e do sistema.

Complexidade

O sistema distribuído é inerentemente complexo, principalmente porque esta arquitectura é sobretudo usada para solucionar problemas complexos. A complexidade da distribuição é provada também pela noção empírica de que um universo articulado por vários núcleos é mais propenso à disfunção do que o universo com um só núcleo. Quando se compartimenta um domínio abdica-se da omnisciência da unidade de controlo, gasta-se energia a comunicar, perde-se o completo controlo na informação, dá-se espaço ao mal entendido, e abre-se a porta á cisão do domínio.

2.3 Porquê Construir Sistemas Computacionais Distribuídos

A essência da resposta é muito simples: porque as pessoas são distribuídas e porque a informação é distribuída.

A adaptação do sistema aos requisitos é um motivo suficientemente forte para o desenho e construção de sistemas distribuídos. Muita vezes, a dispersão geográfica do sistema é tão grande, que restam poucas alternativas aos arquitectos dos sistemas. Contudo este motivo está longe de ser o único que abona a favor desta solução.

Paradoxalmente a maioria das desvantagens citadas em cima podem ser convertidas em vantagens com uma arquitectura e construção correctas.

É incontornável a "mossa" que um canal lento, propenso ao erro e à falha faz no desempenho de um sistema distribuído. Todavia, o facto dos vários componentes e canais de comunicação falharem de forma independente, pode ser aproveitado através da introdução de redundância, para se obter um sistema distribuído mais tolerante à falha, logo mais disponível, do que o seu sistema concentrado equivalente.

Como foi dito, um sistema distribuído é potencialmente inseguro, mas como nenhum componente suporta toda a informação do sistema e todos os métodos para a tratar, uma única brecha no escudo protector normalmente não é suficiente para a extracção de informação coerente que comprometa a segurança de todo o sistema. A dispersão da informação e da via para a usar, pode ser utilizada para criar um sistema distribuído menos vulnerável a ataques, do que o seu sistema concentrado equivalente.

O facto de um sistema computacional distribuído ser normalmente constituído por blocos com funções semelhantes, mas hardware e software de origem diferentes, gera problemas de compatibilidade e de manutenção. Todavia, quando compatibilizados, a

soma dos diferentes componentes do sistema é mais tolerante a defeitos e é mais difícil de atacar.

A sobrecarga de trabalho que deriva da distribuição é inegável, mas por regra o somatório dos componentes de um sistema distribuído é muito mais redundante, potente e paralelo do que o seu equivalente concentrado. Este facto pode facilmente ser convertido num desempenho superior, especialmente se os algoritmos rentabilizarem o paralelismo e a replicação de hardware.

A complexidade de um sistema computacional distribuído é, por regra, superior, mas hoje existe hardware e software disponível que implementam componentes, ferramentas e protocolos altamente fiáveis, que simplificam a distribuição de qualquer sistema.

A organização de um sistema em muitos componentes conduz inevitavelmente a uma maior modularidade do sistema computacional. Por causa disto, este paradigma abre portas a uma fácil expansibilidade do sistema e uma maior reutilização dos seus componentes.

A partir do momento em que se multiplicam componentes, seria de esperar um custo maior. Contudo a economia de escala e a produção em massa, operam "milagres" de redução de custos, o que faz com que hoje muitos pequenos computadores sejam mais baratos do que um grande. Hoje um computador pessoal não é muito menos potente do que um núcleo de um super computador, mas custa muito menos, ou seja em vez de termos um super computador com muitos núcleos, podemos ter pelo mesmo preço muitos mais pequenos computadores. O balanço, em teoria, é a favor do sistema distribuído no que toca ao custo.

Capítulo 3

Arquitectura da *Referee Box* 2008

Neste capítulo apresenta-se o projecto e a implementação da *Referee Box* 2008, designação dada ao sistema de apoio à arbitragem desenvolvido no âmbito deste trabalho.

Começa-se pela análise de requisitos (secção 3.1). A seguir faz-se a apresentação do sistema como um todo (secção 3.2), descrevendo a sua estrutura em rede (secção 3.4) e o protocolo de comunicação entre os nós dessa rede (secção 3.3).Finalmente, apresenta-se a arquitectura interna de alguns nós desenvolvidos, com destaque para o nó central, designado de núcleo (3.5). Os nós periféricos são apresentados na secção 3.6.

Uma parte significativa da *Referee Box* 2008 foi modelada através do método SLiM [20] (SIAS [26]). O modelo e alguma documentação deste (relatórios de especificação SLiM) existem e cobrem sobretudo a definição de requisitos, casos de utilização (CaU) e modelo de conceitos. Estes documentos têm uma importância vital na óptica do programador(es) e criador(es) do sistema, contudo são demasiado temporários, pormenorizados e requerem conhecimentos de modelação e da metodologia usada para serem uma mais valia para o leitor. Optou-se por ilustrar esta dissertação com diagramas simplificados do modelo visado e sempre que as versões pormenorizadas acrescentarem alguma profundidade relevante ao assunto. serão anexas a esta dissertação.

3.1 Requisitos da *Referee Box* 2008

Os requisitos da *Referee Box* 2008 derivam das orientações oficiais do RoboCup [18], do estudo do sistema oficial (*Referee Box* 2005) e do sistema concorrente (*Referee Box* 2007), de várias entrevistas informais com pessoas da área e da experiência pessoal adquirida a operar a *Referee Box* em vários jogos de futebol robótico da M.S.L..

Pode-se dividir estes requisitos em três categorias: funcionais, de desempenho e de segurança.

Apesar dos requisitos funcionais da *Referee Box* 2008 (enumerados na tabela 3.1) se basearem nos requisitos do sistema oficial, deu-se especial relevo à ampliação da modularidade, ao aumento do número de eventos integrados, ao acréscimo do número de actores que operam o sistema, assim como à instalação do multi-controlo e da multi-monitorização remota.

Ref.	Req. funcionais	Restrições
R0.1	Ligar um ou mais árbitros às estações base das equipas.	Rede - TCP/IP sobre Ethernet Rede - Suportar re-conexões
R0.2	Exibir informação de jogo ao público que está no estádio.	Interface - Móvel, Múltipla, Gráfica Tempo resposta - Poucos s
R0.3	Exibir informação de jogo aos elementos humanos das equipas e suas estações base.	Interface - Móvel, Múltipla, Gráfica Tempo resposta - Poucos ms
R1.1	Registar e transmitir os eventos de início e fim do jogo e as suas partes	Interface - Móvel, Múltipla, Gráfica Tempo resposta - Poucos ms
R1.2	Registar e transmitir os eventos de paragem e recomeço do jogo	Interface - Móvel, Múltipla Tempo resposta - Poucos ms
R1.3	Registar as paragens e a reactivações do relógio de jogo	Interface - Móvel, Múltipla Tempo resposta - Poucos ms
R2.1	Registar e transmitir os eventos de substituição, reparação, falta, cartão e golo	Interface - Móvel, Múltipla, Gráfica Tempo resposta - Poucos ms
R3.1	Registar os sub-sistemas e utilizadores	Dados - Tempos, nº conexões, localização
R4.1	Permitir consultar o histórico de jogo em tempo real: o estado, tempos, composição das equipas, etc...	Interface - Móvel, Múltipla, Gráfico
R4.2	Permitir consultar em tempo real informações sobre os sub-sistemas e utilizadores	Interface - Móvel, Múltipla, Gráfica
R5.1	Introduzir dados no início do jogo	Interface - Móvel, Gráfica
R5.2	Criar folha de jogo no fim	Interface - Móvel, Gráfica
R5.3	Iniciar e terminar um jogo remotamente	Interface - Móvel, Gráfica
R5.4	Retomar um jogo abortado	Interface - Móvel, Gráfica

Tabela 3.1: Tabela com requisitos da *Referee Box* 2008.

Os requisitos de desempenho do novo sistema baseiam-se no aumento da fiabilidade, robustez e tolerância à falha em relação ao sistema oficial. Existem constrangimentos temporais em todo o sistema, inclusive no sistema de armazenamento e consulta de informação, porque a interacção entre actores e sistema é feita em tempo real, ou seja em série com o fluxo de jogo.

Os requisitos de segurança da *Referee Box* estão enumerados na tabela 3.2.

A garantia do cumprimento destes requisitos de segurança reside na capacidade de rejeição de eventos e registos incoerentes, assim como num sistema de *logs* central e transversal. Este sistema de registo e *log* serve também de meio dissuasor de violação das regras impostas, na medida em que tudo fica registado e guardado. Na eventualidade da ocorrência de alguma anomalia, facilmente se identifica o prevaricador e se

Todos os módulos são registados
Eventos de registo sem autenticação e sem coerência não são integráveis.
Eventos de jogo, controlos de tempo e eventos de administração só são integráveis se forem produzidos por um módulo registado.
Consultas de informação são válidas se forem produzidos por módulos registados.

Tabela 3.2: Tabela com requisitos de segurança da *Referee Box* 2008.

age em conformidade.

A *Referee Box* 2008 terá de ser um sistema G.P.L. (*General Public Licence*[11]). Os blocos não gráficos terão de poder correr em sistemas embedded, como algum access points, routers, etc...

3.2 Visão Geral da Arquitectura da *Referee Box* 2008

Nas primeiras fases do projecto definiu-se que o sistema iria ser distribuído numa rede. Essa característica foi a solução para algumas das limitações da *Referee Box* oficial. Esta solução foi inicialmente proposta e publicada no artigo [8]. Esta dissertação assenta sobre a concretização, expansão e o aperfeiçoamento da arquitectura proposta nesse artigo.

A principal diferença entre a *Referee Box* 2008 e as actuais *Referee Box*s 2005 e 2007 é a modularidade. A *Referee Box* 2005 e 2007 são dotadas de uma arquitectura mono-bloco, que se limita a ligar duas equipas e um operador através de uma rede. Em contraste com isto, a *Referee Box* 2008 pretende ser um sistema multi-controlado, multi-monitorizado, e geograficamente disperso.

Esta arquitectura justifica-se sobretudo pela dispersão geográfica dos intervenientes, pela crescente necessidade de interacção entre eles, assim como pela sua multiplicidade. Esta característica também permitirá uma fácil inclusão de novas funcionalidades.

Para atender aos requisitos da tabela 3.1 foi proposta uma topologia de rede em estrela, em que o elemento central é o núcleo do sistema, denominado de ***Referee Box* Server Application**. O modelo de computação distribuída adoptado foi o cliente - servidor, onde o elemento central da estrela é o servidor e os elementos periféricos os clientes.

Os nós da rede são aplicações independentes, sendo a comunicação com o servidor realizada pela troca de mensagens numa linguagem baseada em XML. A comunicação é suportada por ligações TCP/IP.

É importante clarificar desde já a diferença entre módulo cliente do sistema e aplicação do sistema. Uma aplicação do sistema é um programa executável e instanciável em qualquer nó da rede. Um módulo cliente do sistema é uma biblioteca, parte de programa ou interface, que implementa um fluxo de dados que é tipificável

como uma peça da arquitectura deste sistema. Uma aplicação cliente pode usar um ou mais módulos cliente do sistema.

No diagrama UML da figura 3.1 estão os blocos principais da arquitectura da *Referee Box* 2008, assim como uma parte da estrutura da aplicação servidora (**Server Application**), nomeadamente as suas três interfaces (Viewer Port, Team Port, Control Port). Em resultado da filosofia cliente - servidor deste sistema, a cada uma destas interfaces corresponderá um porto de escuta na rede (aquele a que algo se conecta - restrição { srv }).

3.2.1 Módulos Cliente do Sistema

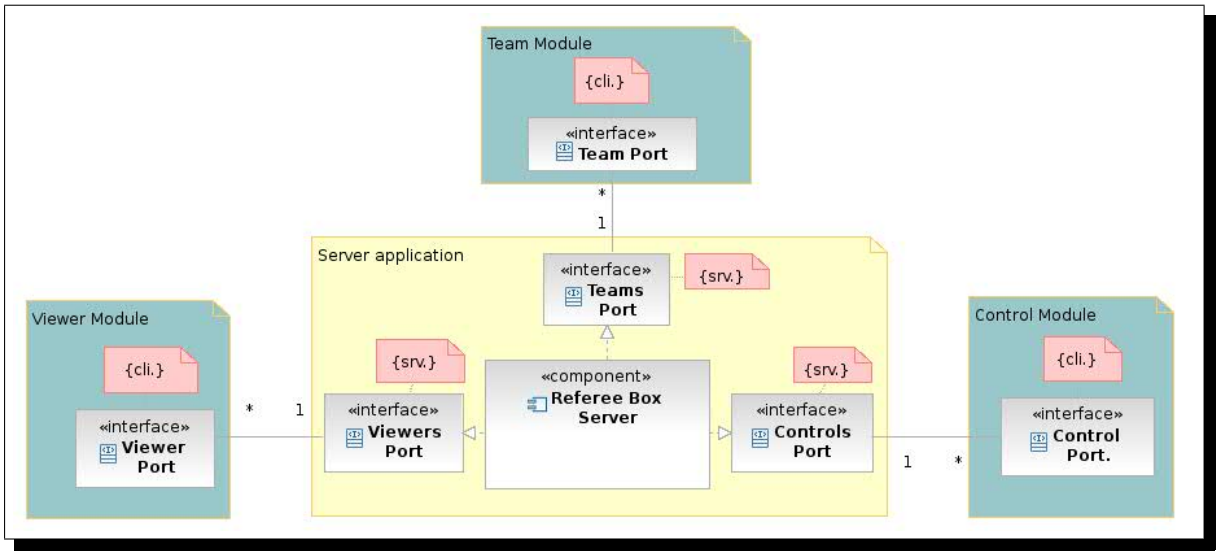


Figura 3.1: Diagrama com os principais componentes *(**UML** component diagram)* da arquitectura da *Referee Box* 2008.

No diagrama da figura 3.1 pode-se verificar também que à volta do núcleo "orbitam" os módulos periféricos ou módulos clientes do sistema (Viewer Module, Team Module, Controls Module). A instanciação destes módulos conduz à criação de conexões de rede ou portos de conexão (aquele que se conecta a algo - restrição { cli }). Cada um destes portos tipifica e compartimenta um fluxo de dados na rede.

É relevante reter que todos os módulos periféricos ou clientes apresentados no diagrama da figura 3.1, podem ser várias vezes instanciados em qualquer nó da rede, isso é modelado através das multiplicidades das ligações entre as interfaces servidor - cliente. É também relevante dizer que todas as comunicações são bi-direccionais.

Control Module

Este módulo implementa uma interface cliente destinada à introdução de informação no sistema. Em cada evento de jogo é gerada e enviada para o servidor uma mensagem por esta via.

A interface do lado do servidor envia periodicamente um relatório com a descrição de todos os módulos periféricos ligados ao núcleo.

Viewer Module

Este módulo implementa uma interface cliente destinada a retirar informação de jogo do sistema. Periodicamente a interface do lado do servidor gera um relatório que descreve pormenorizadamente o estado do jogo.

Team Module

Este módulo implementa uma interface cliente destinada a retirar informação em tempo real dos eventos de jogo. Sempre que um evento é introduzido e integrado no sistema, a interface do lado do servidor envia uma mensagem por esta via.

3.2.2 Aplicações do Sistema

No diagrama da figura 3.2 são apresentadas todas as aplicações que definem a arquitectura da *Referee Box* apresentada no âmbito desta dissertação, e os respectivos módulos. Este diagrama é uma extensão do diagrama de componentes da figura 3.1.

Neste diagrama é claro a existência de aplicações do sistema que usam mais do que um módulo cliente. Por exemplo, a aplicação *Referee Box Command* usa o *Control Module* para fazer chegar as ordens do árbitro ao servidor e usa o *Viewer Module* para receber os dados que lhe permite mostrar o estado do jogo.

O fluxo de mensagens mais significativo deste sistema passa pelo envio de mensagens dos módulos de controlo (*Control Module*) para o núcleo (*Server Application*). Depois disto, o núcleo processa, integra e envia novas mensagens para os módulos de equipa (*Team Module*). O núcleo é também responsável por enviar periodicamente mensagens com estado do jogo para os módulos de visualização (*Viewer Module*) e mensagens com estado do sistema para os módulos de controlo (*Control Module*).

Aplicação Servidora

Server Application

O núcleo do sistema tem a função de conectar todos os módulos periféricos. Recebe, processa, responde e encaminha todas as mensagens que fluem no sistema.

A aplicação servidora ou núcleo (*Server Application* da fig. 3.2) implementa três interfaces de rede, uma interface servidora para cada tipo de módulo cliente. Esta aplicação gera um registo em tempo real com o somatório dos eventos integrados. A informação reunida é depositada (em tempo real) num ficheiro XML (ver secção 3.3). Este ficheiro foi catalogado como a quarta interface do servidor.

A informação reunida nesta aplicação é também a base para a produção e envio de mensagens periódicas com a descrição do sistema e do jogo, enviados via *Control Module* e *Viewer Module*.

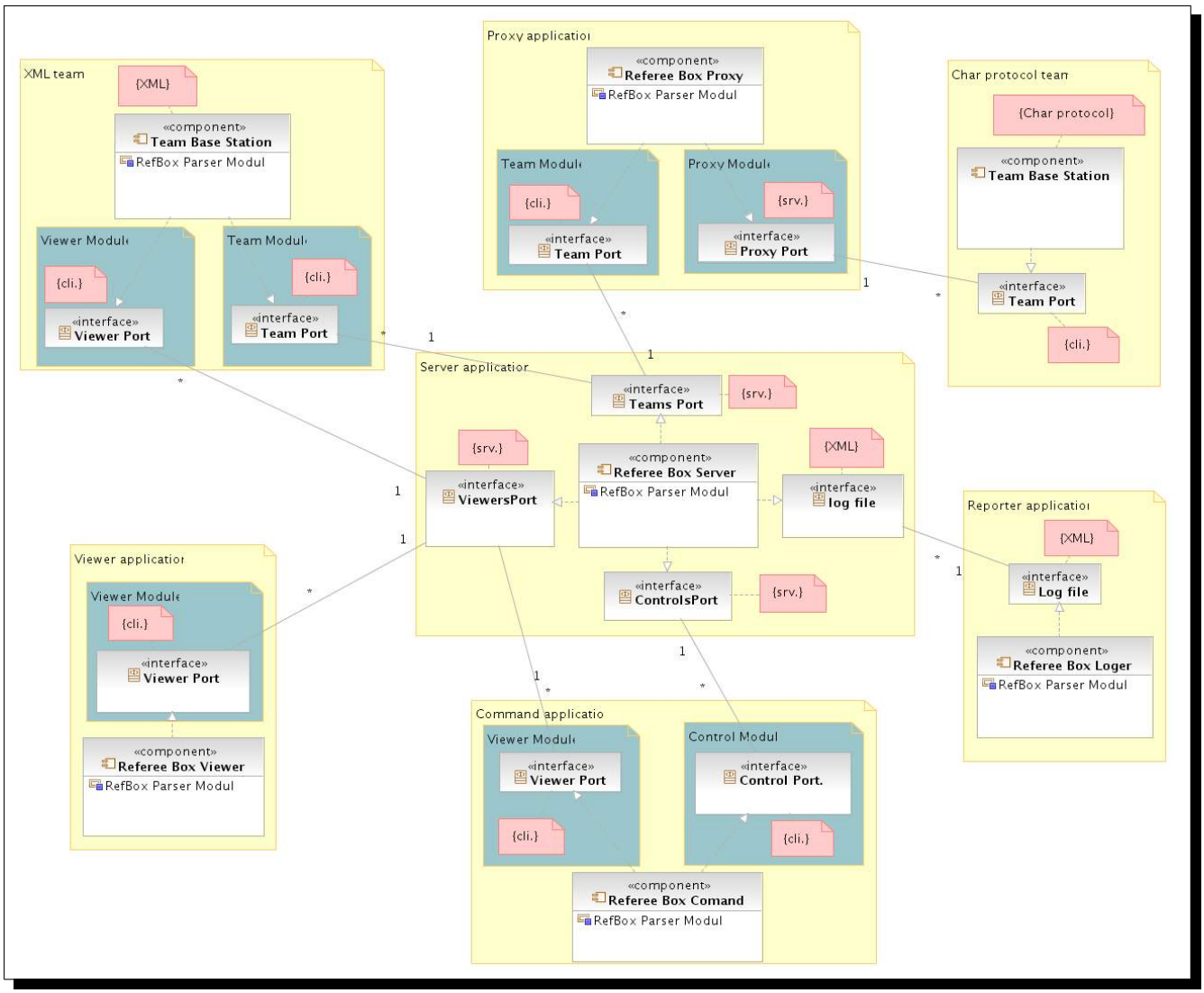


Figura 3.2: Diagrama com todos os componentes *(**UML** component diagram)* da arquitectura da *Referee Box* 2008.

Aplicações Periféricas

O conceito de *Referee Box* apresentado até aqui é genérico e desprovido de orientação no que toca à implementação dos seus blocos. Todavia como esta é uma dissertação num campo eminentemente prático, para provar a exequibilidade do nosso conceito, as cinco aplicações apresentadas na figura 3.2 foram implementadas com um grau de funcionalidade muito próxima dos requisitos apresentados neste capítulo (tabela 3.1).

É importante sublinhar que as aplicações periféricas apresentadas em baixo não esgotam o potencial do conceito. A configuração escolhida pretende criar uma *Referee Box* funcionalmente semelhante às existentes, mas aplicando um paradigma completamente novo.

Command Application

A aplicação de comando (*Command Application* da fig. 3.2) implementa uma interface gráfica, que permite a interacção remota entre um árbitro auxiliar (*Time Keeper*) e o núcleo (*Server Application*). O *Referee Box Command* recorre a dois módulos de sistema, um *Control Module* e um *Viewer Module*. Com eles estes módulos esta aplicação implementa um sistema bi-direccional de controlo e monitorização do servidor.

Proxy Application

Esta aplicação (*Proxy Application* da fig. 3.2) implementa a retro compatibilidade. Para o efeito, traduz as mensagens XML que fluem na rede da *Referee Box* 2008 para os caracteres do protocolo em vigor.

Quando esta aplicação é usada, substitui o núcleo (*Server Application*) do ponto de vista da equipa, ou seja, implementa uma ponte entre o "novo" servidor e a equipa "antiga".

Esta aplicação emula uma equipa no sistema usando o seu módulo (*Team Module*) e capta todas as mensagem a ela dirigida. No caso de ser possível traduzir a mensagem envia-a à equipa real conectada no seu servidor (*Proxy Module*).

Viewer Application

Esta aplicação (*Viewer Application* da fig. 3.2) exibe informações de jogo, como jogadores em campo, resultado, tempo, período, assim como publicidade e animações destinadas ao público que assiste a uma partida. Esta aplicação não deve ser confundida com o módulo de sistema com o mesmo nome (*Viewer Module*). Apesar deste módulo ser a base desta aplicação, esta é apenas um dos possíveis usos deste módulo de sistema.

Reporter Application

Esta aplicação (*Reporter Application* da fig. 3.2) lê o *log* produzido pelo núcleo (*Server Application*) e gera um documento escrito em LaTeX [1] . Este documento TeX [2] pode ser compilado num relatório de jogo legível (PDF ou num web site HTML).

3.3 Protocolo da *Referee Box* 2008

Todas as mensagens que transitam entre aplicações do sistema são escritas numa linguagem textual baseada na linguagem XML [3]. O mesmo acontece com os registos no ficheiro de log. A descrição desta linguagem pode ser encontrada no anexo A.

Todas as aplicações que recebem mensagens nesta linguagem precisam de as analisar sintacticamente e convertê-las para uma estrutura de dados adequada. Construiu-se um

[1] O LaTeX é um conjunto de macros para o processador de textos TeX. Por causa de sua alta qualidade tipográfica, é usado sobretudo para a produção de textos matemáticos e científicos, nomeadamente: artigos, livros e teses. Esta dissertação foi escrita em LaTeX.

[2] O TeX (pronuncia-se "tech"da palavra grega "*téchne*", arte) é um compilador tipográfico popular no meio académico. Normalmente não se usa TeX directamente, usa-se LaTeX

[3] XML *(eXtensible Markup Language)* é uma recomendação da W3C *(World Wide Web Consortium)* para gerar linguagens de marcação para necessidades especiais.

reconhecedor sintáctico ou *parser* [4] único para o fazer. Esta funcionalidade foi integrada no módulo *Refbox Parser Module* que, por isso, aparece em todas as aplicações (ver figura 3.2). O reconhecedor sintáctico foi escrito em flex [5].

3.4 Rede da *Referee Box* 2008

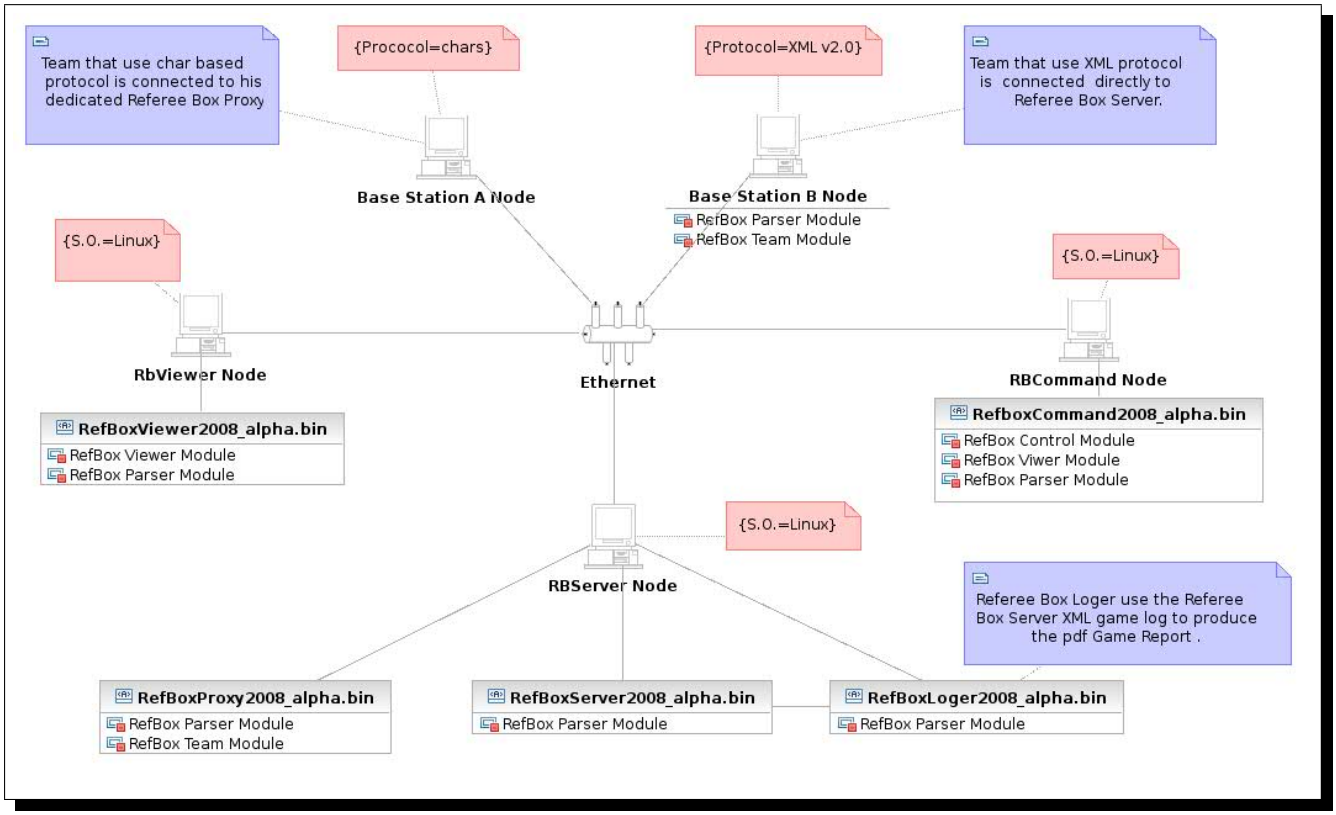


Figura 3.3: Diagrama a instalação *(**UML** deployment diagram)* da *Referee Box* 2008.

Segundo a classificação do modelo OSCI [6] a rede que conecta os vários blocos da *Referee Box* 2008 é uma rede com ligação de dados Ethernet [7] , com o protocolo de transporte TCP [8] , com protocolo de rede IP [9] e com a apresentação em caracteres

[4]Em inglês o termo *parsing* corresponde a análise sintáctica.

[5]Flex *(The Fast Lexical Analyzer)* é uma gerador de análise lexical rápido. É uma ferramenta para criar programas de reconhecimento de padrões em texto. O Flex é uma implementação não livre do analisador lexical livre Lex.

[6]OSI é um modelo ISO *(International Organization for Standardization)* que permite a comunicação entre máquinas heterogéneas.

[7]Ethernet é um esquema de partilha de canal denominado CSMA/CD *(Carrier Sense Multiple Access with Collision Detection)*.

[8]TCP *(Transmission Control Protocol)* é um protocolo de transporte, orientado à conexão, ponto a ponto, confiável, full duplex, com handshake e controlo de fluxo.

[9]IP *(Internet Protocol)* é um protocolo de encaminhamento que garante a identificação de todos os pontos da malha da rede.

ASCII que definem mensagens no protocolo de aplicação XML.

A *Referee Box* 2008, na sua configuração actual, é constituída por cinco aplicações distintas. Cada uma delas é completamente independente, na medida em que pode correr em qualquer nó da rede.

Na instalação do diagrama da figura 3.3 o *Referee Box* Proxy, o *Referee Box* Server e o *Referee Box* Reporter correm no mesmo nó - R.B. Server Node.

Quando existe apenas um operador do sistema, é normal a fusão do *R.B. Server Node* e do *R.B. Command Node*. Esta configuração definem uma *Referee Box* 2008 minimalista, que se aproxima muito da configuração do sistema oficial (*Referee Box* 2005). Esta concentração na *Referee Box* 2008, pode-se justificar em algumas ocasiões por questões práticas, relacionadas com a operação das aplicações, bem como com a racionalização de recursos.

O diagrama de instalação da figura 3.3 representa a configuração normal, não a única. A modularidade do sistema permite inúmeros tipos de instalações. É responsabilidade da equipa de arbitragem escolher a que melhor se adapta à situação em causa.

As estações Base das equipas fazem parte do sistema e da rede de jogo, na mediada em que este seria inútil sem elas, não sendo, contudo, partes da *Referee Box*. Não obstante isto, as equipas implementam um módulo de sistema (*Team Module*) e podem usar nas suas estações base algumas bibliotecas da *Referee Box* 2008, nomeadamente a biblioteca de *parsing* das mensagens XML do sistema.

3.5 Aplicação Servidora

Destacam-se a seguir as opções mais importantes que se fizeram no que respeita à estrutura do núcleo do sistema. Todavia, é de realçar que daqui em diante, a estrutura de software apresentada é uma selecção do universo de modelos que este conceito de *Referee Box* distribuída contempla.

Optou-se por apresentar em mais pormenor o núcleo porque a maioria das características do sistema propaga-se a ele e porque é o bloco mais estruturante e exemplificativo do sistema.

3.5.1 *Referee Box Server Application*

A arquitectura da aplicação servidora (núcleo do sistema) é apresentada num modelo criado através de uma metodologia baseada no método SLiM [20] (SIAS [26]). Para chegar a esta arquitectura e modelo começou-se por definir requisitos na óptica do utilizador através de um modelo de casos de utilização (*CaU*); em seguida, tendo por base este modelo de *CaU*, os requisitos da tabela 3.1, a distribuição e a modularidade escolhida, desenvolveu-se o modelo de conceitos, o modelo lógico.

A apresentação desta secção baseia-se sobretudo no modelo lógico da aplicação, ou seja, no modelo que descreve a implementação e a arquitectura da aplicação. Neste modelo figuram todas as classes de software e suas interacções. Contudo, em prol da

objectividade e simplicidade, estas classes serão aqui apresentadas de uma forma muito genérica e simplificada.

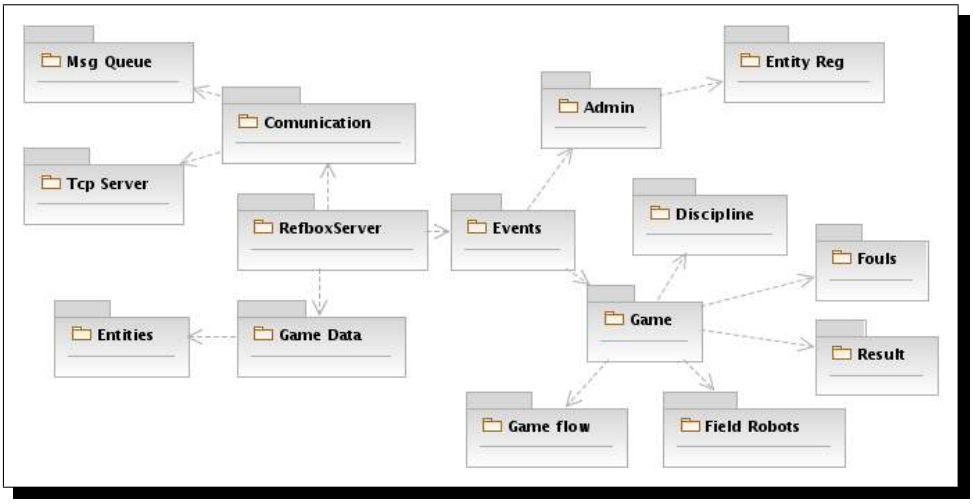


Figura 3.4: Diagrama pacotes (UML package diagram) do modelo lógico.

Empacotamento de classes

O empacotamento das classes do modelo lógico (figura 3.4) deriva do empacotamento do modelo de conceitos e representa a estrutura da aplicação. A estrutura do modelo lógico define também a árvore de pastas que contém os ficheiros de código fonte.

O núcleo da *Referee Box* 2008 foi orientado ao objecto e modelado em UML [10]. Como se pode antever na forma do diagrama de pacotes da figura 3.4, o núcleo da *Referee Box* 2008 é "sediado" num pacote central (*package Refbox Server*) que depende do pacote "Eventos" (*package Events*), do "Comunicações" (*package Communications*), do "Entidades" (*package Entitis*) e do pacote "Registo de jogo" (*package Game Data*).

O pacote de "Comunicações" (*Comunication* - fig. 3.4) agrega as classes que implementam a comunicação (fig. 3.6(b)). Este pacote divide-se no pacote de comunicação externa (*package Tcp Server*), para as comunicações entre módulos **através da rede**, e no pacote de comunicação interna (*package Msg Queue*), para a comunicação entre threads [11] **através da fila de mensagens**.

As classes que moldam o registo de jogo (fig. 3.6(d)) fazem parte do pacote com o mesmo nome (*package Game data* - fig. 3.4).

As classes de evento, definidas no modelo hierárquico da figura 3.5, fazem parte da árvore de pacotes apresentada no lado direito do diagrama da figura 3.4.

[10] Unified Modeling Language [10] é uma linguagem ISO de modelação de software e sistemas de informação não proprietária.

[11] *Thread*, em português "fio de execução", é uma abreviatura para *thread of execution*. Os threads são uma das vias para bifurcar ou dividir a execução de um Processo ou programa em duas ou mais tarefas pseudo-simultâneas.

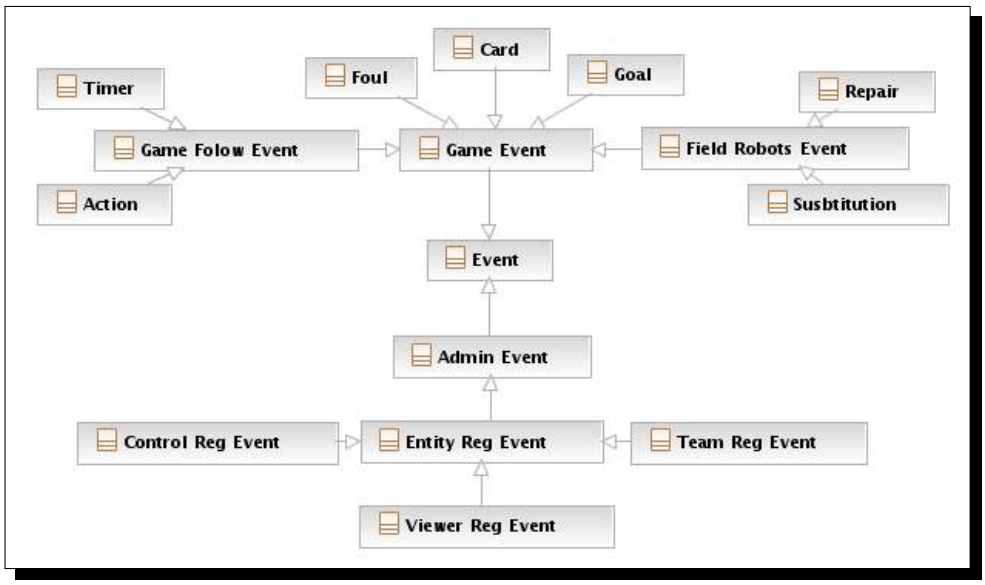


Figura 3.5: Hierárquico de classes de evento (UML class diagram).

Estrutura dos Objectos e Classes

Como se pode verificar no diagrama da figura 3.6(c), a instanciação do objecto contentor de todo o núcleo (*server*) implica a criação de vários objectos de rede (*srv*), um objecto com o registo de jogo (*game registry*) e um objecto que implementa uma fila de mensagens (*msg srv*). Cada um dos objectos de rede (*srv*), por sua vez, instancia um objecto (*msg cli*) que deposita mensagens na fila previamente criada.

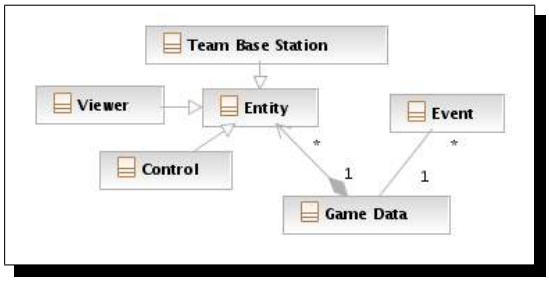
É relevante sublinhar que a fila de mensagens e a interface de rede aqui apresentada são componentes do modelo, não são pormenores de implementação. Estes componentes foram construidos com recursos do sistema operativo, pelo que dependem deste. Todavia, em abono da portabilidade e modularidade, também foram encapsulados (classes *TCP Server, Msg Queue Cli e Msg Queue Srv* - fig. 3.6(b)).

O núcleo da *Referee Box* 2008 é uma aplicação de funcionamento em tempo real orientada ao evento. Os objectos de evento (*event* - fig. 3.6(c)) definidos na hierarquia de classes de evento (*base class Event* - fig.3.5) são criados pelo objecto contentor (*server*) à custa das mensagens de rede que chegam através dos objectos (*srv*). Cada objecto de evento é um vector de modificação do registo do jogo (*game registry*),

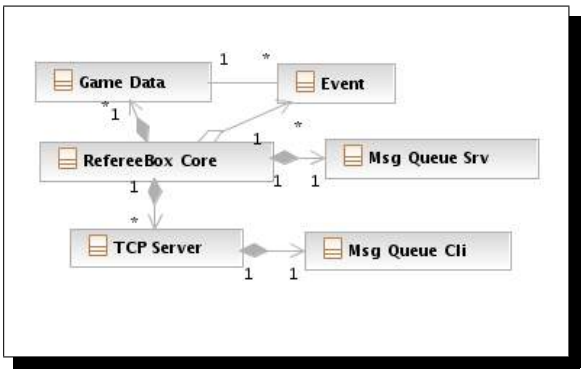
O hierárquico de classes Entidade (*base class Entity* - fig.3.6(a)) faz parte do registo do jogo, e define os objectos que contêm dados dos actores e sistemas ligados ao núcleo da *Referee Box*.

Denomina-se também de "Registo de jogo" a classe (*Game Data*) que define o objecto (*game registry*) cujos dados e relações definem um jogo num determinado momento. Este objecto é o único interface do registo de jogo.

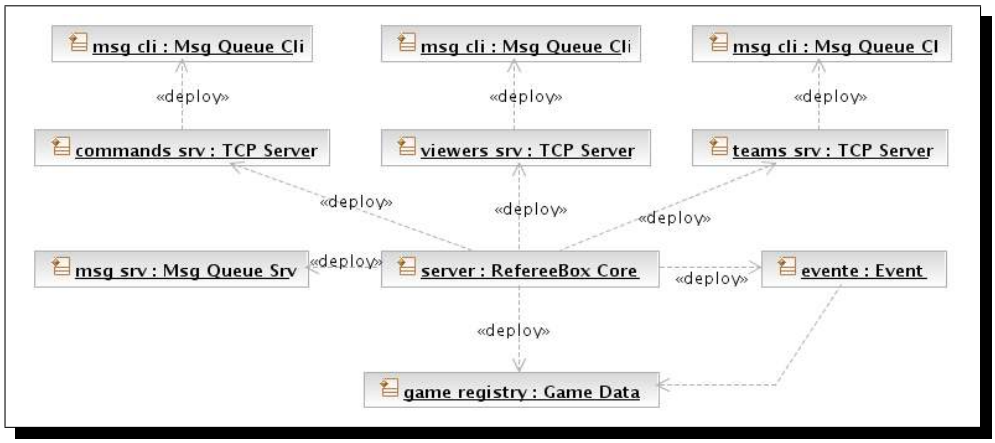
Um objecto do modelo hierárquico da figura 3.5 encerra em si toda a informação de um evento e a via para o integrar no registo de jogo. Para o efeito, o objecto de evento (*event*) manipula uma ou mais funções do objecto interface do registo de jogo (*game registry*) que, por sua vez, cria e manipula outros objectos que o auxiliam a



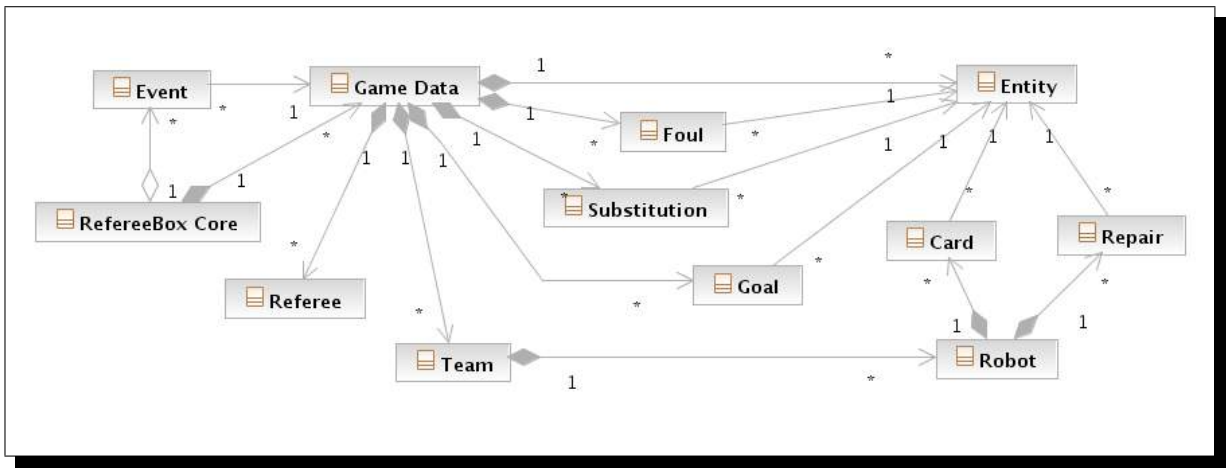
(a) Diagrama de classes (*UML class diagram*) de entidades e suas relações.



(b) Diagrama de classes (*UML class diagram*) de comunicação e suas relações.



(c) Diagrama objectos (UML object diagram) que definem a estrutura do *Referee Box Server 2008*.



(d) Diagrama de classes (*UML class diagram*) do registo de jogo (*Game Data*) suas colecções (*Team*, *Referee*, *Foul*, *Substitution*, Goal, *Entity*).

Figura 3.6: Diagramas principais do modelo do núcleo da *Referee Box* 2008.

compartimentar e gerir os dados de jogo.

O registo de jogo é o somatório do seu interface, definido pela classe *Game Data*, e de um conjunto de colecções, definidas pelas classes: *Entity*, *Foul*, *Substitution*, *Goal*, *Team* e *Referee* (fig. 3.6(d)). Pode-se também verificar no diagrama dessa figura que o objecto *Team* cria e manipula uma colecção de *Robot(s)*, este, por sua vez, cria e manipula uma colecção de *Card(s)* e *Repair(s)*.

Dinâmica dos Objectos

As peças da estrutura apresentada em cima interagem sempre no sentido de propagarem a ocorrência de um evento. O ponto de partida é a recepção de uma mensagem. Em seguida são actualizados os registos, criados os relatórios e emitidas as mensagens. As dezenas de eventos integrados neste núcleo tornam irrazoável a sua exaustiva descrição nesta dissertação, motivo pelo qual, se seleccionaram os fluxos mais exemplificativos da dinâmica do sistema para o apresentar e justificar a estrutura anteriormente descrita.

A partir das sequências, apresentadas no anexo C, é possível verificar que todos os eventos, durante o seu processo de integração, passam por quatro passos, entre a sua criação e a sua destruição:

1. Validação semântica
2. Validação no registo do jogo
3. Actualização do registo do jogo
4. Resposta / Reencaminhamento

A classe base de todos os eventos prototipa estas funções virtuais, o que implica que seja responsabilidade de cada evento recarregá-las. Do ponto de vista do objecto agregador de todo o núcleo, todos os eventos são iguais no que toca à sua integração, bastando invocar estas funções sequencialmente.

Paralelismo

Na descrição estrutural deste núcleo foram apresentadas classes que implementam uma fila de mensagens. A existência desta fila é justificada pelo paralelismo da aplicação e pela necessidade de comunicação entre os seus vários fios de execução (*threads*).

A estrutura concorrente do núcleo da *Referee Box* 2008 é apresentada no diagrama de actividades da figura 3.7. Esta estrutura paralela baseia-se num *thread* principal (pai) (*R.B. Core*), onde corre o corpo do programa, e em vários *threads* secundários (filhos), que implementam a interface de rede deste núcleo. No fio de execução principal corre o tipo de acções modeladas nos diagramas das figuras C.3, C.2 e C.1, enquanto que nos threads secundários correm as acções implementadas pelos objectos da classe *TCP Server*.

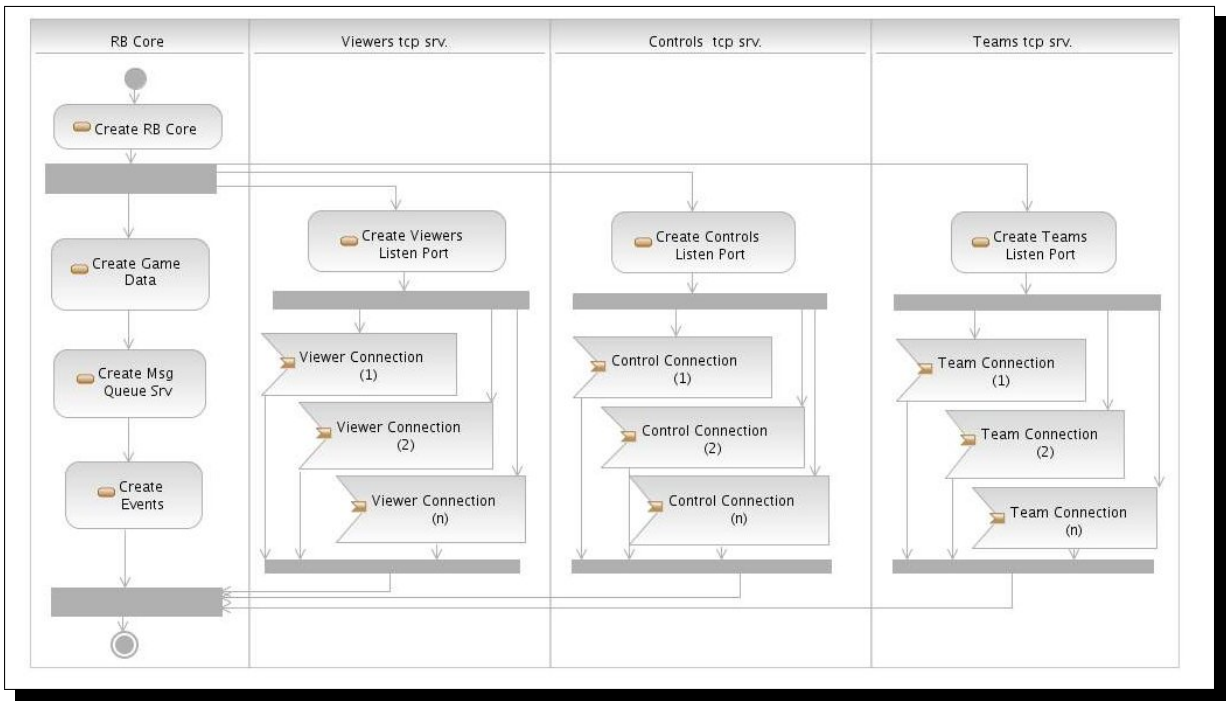


Figura 3.7: Diagrama de actividade (*UML activity diagram*) com a descrição da estrutura concorrente do núcleo da *Referee Box* 2008.

Neste sistema, a fila de mensagens é a única via de comunicação entre *threads* e foi criada sob a paradigma cliente-servidor. Esta catalogação da fila, não muito convencional em IPC [12], deriva do modelo do sistema e serve para transmitir que esta fila tem vários pontos de depósito de mensagens, mas um só de recepção, ou seja, apenas fluem mensagens dos *threads* filhos, que implementam a interface de rede, para o *thread* pai, que suporta o corpo do programa.

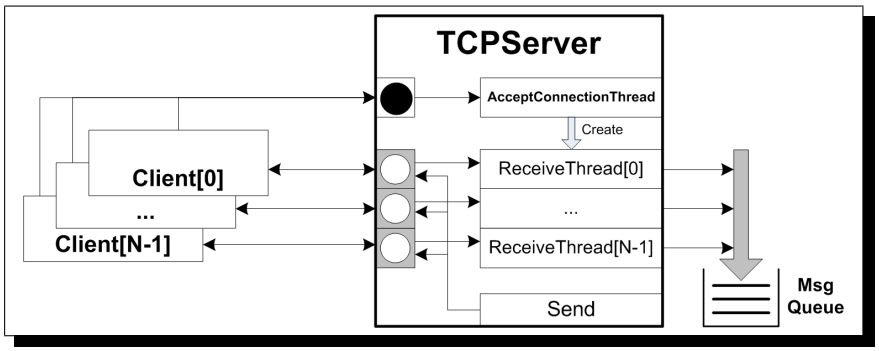


Figura 3.8: Diagrama de blocos com a estrutura concorrente e algumas relações dos objectos da classe *TCP Server*. (Figura retirada do artigo [8]).

Como se pode verificar no diagrama das figuras 3.7 e 3.8, o processo de criação de um objecto interface de rede (*class TCP Server*) desencadeia a bifurcação do fio de

[12] I.P.C. - *Inter Process Comunications*

execução e a criação de um porto de escuta na rede. Quando uma entidade se conecta a esse porto é novamente bifurcado o fio de execução e canalizada esta conexão para um novo porto atendido num novo *thread*.

Nos diagramas da figura 3.8 mostra-se que cada *thread* de conexão é dedicado a uma entidade conectada ao sistema e quando uma mensagem é recebida, é catalogada e inserida na fila de mensagens. O thread principal integra sequencialmente as mensagens colocadas na fila e responde directamente para entidades conectadas ao sistema.

3.6 Aplicações Periféricas

Como já foi referido na secção anterior, a arquitectura das aplicações aqui apresentada não esgota o potencial do conceito. Em abono do rigor, dever-se-ia falar de arquitectura de algumas aplicações periféricas, nomeadamente das aplicações que são uma via para demonstrar a funcionalidade do conceito apresentado neste capítulo.

Uma aplicação periférica é uma aplicação desenvolvida para interagir com a aplicação servidor através do protocolo de comunicações estabelecido. A função da aplicação é da exclusiva responsabilidade do seu projectista e pode recorrer a um ou mais dos interfaces disponibilizados pelo servidor.

Há cinco vocações naturais para este tipo aplicações:

- Estações base das equipas;
- Painéis de comandos que permitem ao árbitro ou seu auxiliar dar ordens ao sistema;
- Mostradores de informação de jogo;
- Pontes de adequação para equipas usando o protocolo antigo; e
- Produtores de relatórios de jogo.

No entanto, outro tipo de aplicação poderá aparecer.

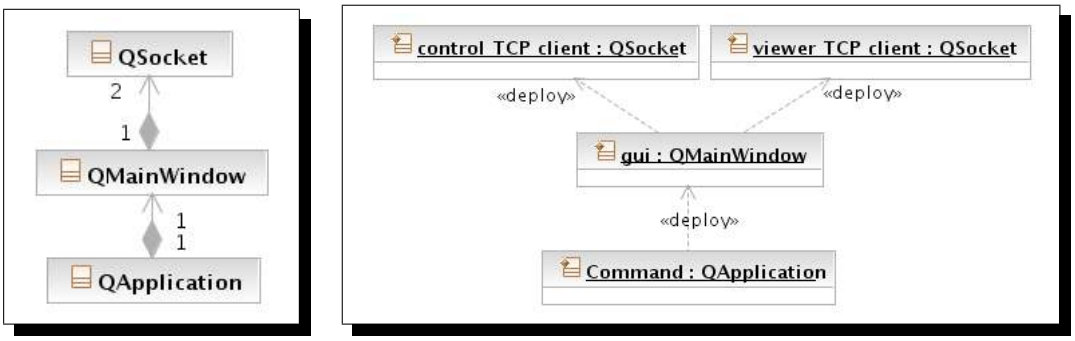
As estações base das equipas são, tipicamente, desenvolvidas pelas próprias equipas. Em relação aos restantes quatro tipos desenvolveu-se um exemplo de cada que são apresentados a seguir.

3.6.1 *Referee Box Command Application*

O *Referee Box* Command é uma aplicação periférica que conjuga as valências painel de comandos e mostrador. Representa um painel de botões, actuados por rato e por teclas, através do qual o árbitro ou o seu assistente enviam ao servidor as ordens de arbitragem. A sequência de passos para uma dada ordem é acompanhada de mudanças visuais que facilitam a operação. Ao mesmo tempo a aplicação mostra o estado no jogo em cada momento.

O *Referee Box* Command é uma aplicação orientada ao objecto que usa os recursos gráficos e de rede do QT [13] . Em contraste com o núcleo, que implementa as suas funcionalidades com chamadas de sistema, esta aplicação apoia-se sobretudo em classes fornecidas pelo QT para simplificar, aumentar a portabilidade e a fiabilidade da aplicação.

A estrutura de classes e objectos simplificada desta aplicação é apresentada na figura 3.9.



(a) Diagrama de classes (*UML class diagram*).

(b) Diagrama objectos (UML object diagram).

Figura 3.9: Diagramas do *Referee Box Command 2008*.

Dado que esta aplicação é exclusivamente gráfica, o objecto contentor da aplicação (*command*) da classe *QApplication* instancia o interface gráfico (*gui*) da classe *QMainWindow*. Por sua vez, este objecto instancia uma multiplicidade de outros, de onde se destacam os objectos de rede (*control e viewer*) da class *QSocket*.

O fluxo básico deste sistema consiste no lançamento da aplicação, seguida do registo dos módulos de rede e da recepção de relatórios, cuja informação serve para actualizar a interface gráfica. Por outro lado, consiste também no envio de mensagens para o módulo de comando mediante a actividade do operador.

3.6.2 *Referee Box Proxy Application*

O *Referee Box* Proxy é uma aplicação periférica que implementa a ponte de adequação para equipas que usam o protocolo oficial. Do ponto de vista do servidor esta aplicação funciona como uma estação base. Do ponto de vista das estações bases das equipas funciona como um servidor do protocolo antigo.

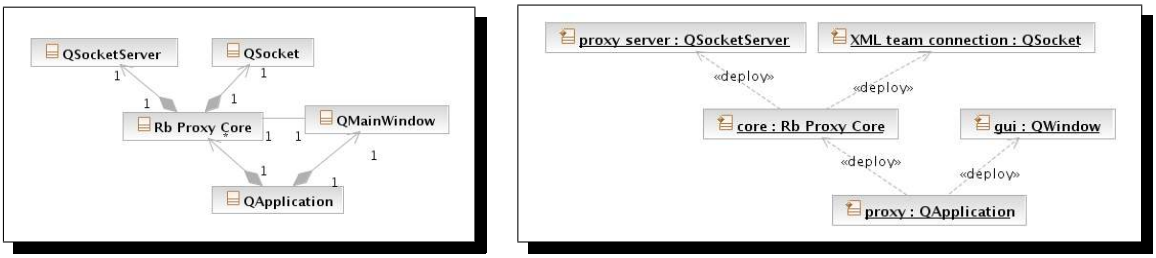
Cada instância desta aplicação é dedicada a uma só equipa e possui dois modos de funcionamento: gráfico e não gráfico.

No processo de ligação de uma equipa é necessário enviar ao *Referee Box* server uma mensagem XML que descreve a equipa. Dado que a equipa ligada ao *proxy* usa o

[13] O QT é uma ferramenta multi-sistema operativo, completamente orientada ao objecto e escrita em C++. Fornece um M.O.C. (Meta Object Compiler), grande número de classes que implementam variados recursos, aplicações amigas do utilizador para desenvolver projectos com interfaces gráficas (QT designer) e gestão de línguas (QT linguistic).

protocolo oficial, não o XML, no processo de emulação da equipa, é responsabilidade do *proxy* recolher e enviar essa mensagem. Para o efeito lê a mensagem de um ficheiro, que lhe é passado pela linha de comandos ou pelo interface gráfico.

O *Referee Box* Proxy também é uma aplicação orientada ao objecto que usa os recursos gráficos e de rede do QT. A estrutura de classes e objectos simplificada desta aplicação é apresentada na figura 3.10.



(a) Diagrama de classes (*UML class diagram*).

(b) Diagrama objectos (UML object diagram).

Figura 3.10: Diagramas do *Referee Box Proxy 2008*.

Dado que esta aplicação pode ou não usar a sua interface gráfica, o objecto contentor da aplicação (*proxy*) pode não instanciar o interface gráfico (*gui*). No entanto, o objecto núcleo desta aplicação (*core*), os objectos de rede (*proxy server* e *xml team connection*) são incondicionalmente instanciados. A interface gráfica (*gui*) comunica com o núcleo desta aplicação (*core*) através de *slots* e *signals* [14] implementados pelo QT.

O fluxo básico deste sistema consiste no lançamento da aplicação, seguida da conexão de uma equipa no servidor do *Referee Box Proxy*, que despoleta a ligação ao núcleo do sistema e o envio de um ficheiro de registo. Depois deste processo as mensagem são recebidas, traduzidas e enviadas à equipa "real".

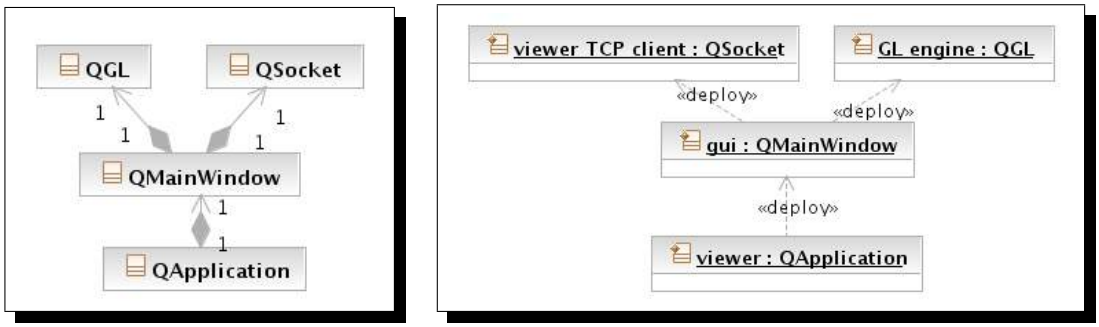
3.6.3 *Referee Box Viewer Application*

O *Referee Box* Viewer é uma aplicação periférica que exibe informações de jogo, como jogadores em campo, resultado, tempo, período, assim como publicidade e animações destinadas ao público que assiste a uma partida.

O *Referee Box Viewer* é um mero *front-end*, pelo que é desprovido de qualquer tipo de processamento relevante ao jogo. A acção deste sistema resume-se ao registo no núcleo e à exibição da informação recebida através dos relatórios XML que este lhe envia periodicamente.

o *Referee Box* Viewer também é uma aplicação orientada ao objecto que usa os recursos gráficos e de rede do QT.

[14] O pré-compilador ou M.O.C. do QT, através do acréscimo de algumas palavras reservadas, enriquecem o C++ com abstracções como o conceito de *slot* e *signal*. Estas estruturas são traduzidas pelo M.O.C. em vulgar código C++. Desta maneira, o código para o QT pode ser orientado ao evento, ou seja, algo acontece, é gerado um *signal* que pode estar conectado a um ou mais *slots*, despoletando assim, uma ou mais acções.



(a) Diagrama de classes (*UML class diagram*).

(b) Diagrama objectos (UML object diagram).

Figura 3.11: Diagramas do *Referee Box Viewer 2008*.

Dado que esta aplicação é exclusivamente gráfica, o objecto contentor da aplicação (*viewer*) da classe *QApplication* instancia o interface gráfico *gui* da classe *QMainWindow*. Por sua vez, este objecto instancia uma multiplicidade de outros, de onde se destacam os objectos de rede (*viewer*) da class *QSocket* e o objecto *GL engine* da classe *QGL*. Este último é responsável por fazer o *rendering* de uma parte da área desta interface gráfica usando a biblioteca gráfica OpenGL[15].

3.6.4 *Referee Box Reporter Application*

O *Referee Box* Reporter é uma aplicação conversora de ficheiros orientada à função. Tem como entrada um ficheiro ASCII, com texto no protocolo XML da *Referee Box* 2008, e tem como saída um ficheiro LaTeX.

O processo de tradução inicia-se com o *parsing* do relatório XML, através do módulo *Referee Box* Parser 2008, em seguida, com essa informação é escrito o ficheiro LaTeX.

O *Referee Box* Reporter não produz um ficheiro directamente compilável com o TeX. Esta aplicação produz um ficheiro com dados que é incluído num *template* com a forma. Quando juntos, estes dois ficheiros, podem ser transformados num relatório passível de ser lido, impresso e publicado.

[15] O OpenGL[21] é uma biblioteca gráfica 2D e 3D.

Capítulo 4

Resultados

Neste capítulo apresentam-se as interfaces, as funcionalidades, e os problemas de cada uma das aplicações desenvolvidas.

4.1 Aplicação Servidora

Os requisitos de todo o sistema, enumerados na tabela 3.1 (pág.22), propagam-se na totalidade à aplicação servidora (núcleo).

4.1.1 *Referee Box* Server 2008

O núcleo do sistema corre exclusivamente em Linux, foi escrito em C++ e tem a função de conectar todos os módulos periféricos: recebe, processa, responde e encaminha todas as mensagens que fluem no sistema.

Esta aplicação, mais precisamente o sistema concorrente e de rede, é fortemente dependente do sistema operativo. Em abono da *performance* estes recursos foram implementados de raiz, através da construção de classes que encapsulam estas funcionalidades e usam abundantemente chamadas de sistema Unix/Linux.

Interface com o Utilizador

A interface com o utilizador do *Referee Box Server 2008* é de linha de comandos e apoia-se num menu de ajuda (tab. 4.1) onde são enumerados o conjunto de parâmetros que podem ser passados no lançamento. No entanto a estrutura apresentada na secção 3.5 foi preparada para ser compatível com o funcionamento de uma interface gráfica capaz de lançar, encerrar, monitorizar e modificar os parâmetros de funcionamento do núcleo (portos, ficheiros de log e registo), assim como para modificar os parâmetros estáticos de jogo (número do jogo e localização/nome do torneio).

Funcionalidades

O *Referee Box server 2008* foi projectado para implementar as funcionalidades enumeradas nas tabelas 4.2 e 4.3. Todavia o protótipo que aqui se descreve implementa

Referee Box Server Bash Help	
Option	Description
PORTS	25000 < Port Range < 35000
-c num	Controls-Port dft=30000
-t num	Teams-Port dft=30100
-v num	Viwers-Port dft=30200
ERRORS	dft=Errors,Logs->file
-E	Errors->console, Logs->file
-V	Errors and Logs -> console
-f srt	Log Folder Name dft = RBServerLog
-p srt	Log Folder path dft=/tmp/
GAME INFO	dftNum=0 dftStr=NoLocal
-l srt	Game Location
-n num	Game Number
-G	Launch Refbox Server G.U.I.
-h	help - this list

Tabela 4.1: A lista de parâmetros pode ser consultada no menu de ajuda do *Referee Box Server 2008* (./RefboxServer2008 -h).

as funcionalidades enumeradas na tabela 4.2.

Implementar três portos de escuta na rede (equipas, comandos e visualizadores).
Implementar ligações de rede independentes para cada entidade conectada ao núcleo.
Obrigar o registo de todas as entidades que introduzem ou retiram informação do núcleo.
Criar um *log* de sistema com todos os erros, avisos e acontecimentos relevantes do núcleo.
Receber dos *Referee Box Control* mensagens XML que descrevem eventos de jogo.
Criar um registo XML com o somatório dos eventos de jogo recebidos.
Enviar para as equipas uma mensagem XML a descrever cada evento de jogo.
Enviar periodicamente para os visualizadores um relatório XML que descreve o estado do jogo.
Enviar periodicamente para os comandos um relatório XML que descreve as entidades registadas no núcleo.
No início de cada jogo, registar os dados estáticos deste.

Tabela 4.2: Funcionalidades implementadas no *Referee Box server 2008*.

Estado e Problemas

O *Referee Box Server 2008* revelou-se uma aplicação funcional nas funcionalidades catalogadas como completas. Nesta configuração o sistema de rede, a integração de eventos, a produção de *logs*, de registos de jogo e de relatórios, estão de acordo com os requisitos de projecto.

Identificar e permitir a re-conexão de uma entidade registada.
Criar e terminar um jogo em resposta a uma mensagem de um *Referee Box Control*.
Retomar um jogo abortado.
Retomar um jogo abortado em resposta a uma mensagens de um *Referee Box Control*.
Implementar uma interface gráfica capaz de lançar, encerrar, monitorizar, modificar parâmetros de funcionamento do núcleo e parametros estáticos de cada jogo.
Permitir o funcionamento e registo de "equipas mistas", ou seja equipas que são a junção de duas ou mais equipas e suas respectivas estações base.

Tabela 4.3: Funcionalidades não implementadas no *Referee Box server 2008*.

O *Referee Box Server 2008* tem vários *bugs* reportados e não corrigidos, nomeadamente nos objectos de rede (*class TCP Server* - secção 3.5). Devido a má gestão do sistema paralelo, estes objectos falham quando duas ou mais entidades se conectam em **simultâneo** ao mesmo porto de escuta. O outro bug reportado consiste na não remoção da fila de mensagens fornecida pelo Sistema Operativo, quando a aplicação falha. O facto desta fila já existir impede o relançamento do núcleo pelo que o utilizador tem de a eliminar manualmente na recuperação de uma falha.

Não obstante os enumeros testes pelos quais esta aplicação já passou, considera-se que não estão esgotados, mais concretamente as funcionalidades estáveis deverão ser testadas por pessoas não relacionadas com a sua construção.

4.2 Aplicações Periféricas

4.2.1 *Referee Box* Command 2008

O *Referee Box Command* é uma aplicação escrita em C++/QT3 sobre Linux, que permite a interacção remota entre um árbitro auxiliar (*Time Keeper*) e o núcleo (*Server Application*).

Na *Referee Box* 2008, o *Command* é o único meio de introduzir dados no sistema. Todavia esta aplicação pode ser multi-instanciada em qualquer nó da rede, ou seja vários *Commands* podem operar simultaneamente sobre o mesmo *Referee Box Server*.

O *Referee Box Command* é um mero *front-end*, pelo que é desprovido de qualquer tipo de processamento relevante ao jogo. A acção do utilizador sobre esta interface produz mensagens XML dirigidas ao servidor. Este, por sua vez, envia periodicamente relatórios a partir dos quais este comando actualiza a informação exibida e os botões activos.

O aspecto deste comando foi claramente inspirado na actual interface da *Referee Box* oficial. Contudo introduziram-se algumas novidades para aumentar a ergonomia e usabilidade do sistema, nomeadamente mais informação integrada na interface, a utilização guiada, o modo de introdução de eventos com teclas e rato em simultâneo e o "teclado" de jogadores com informação embutida.

Interface com o Utilizador

Esta aplicação implementa uma interface gráfica com o utilizador (*G.U.I.* [1]) apresentada nas figuras 4.1 e 4.2.

Este comando foi desenhado para fornecer a um ou vários árbitros uma via para administrar o sistema (*frame admin*), controlar o servidor (principalmente no *frame Control*) e monitorizar os parâmetros de jogo (principalmente no *frame Viewer*). Fora destes *frames* são sempre exibidas as informações críticas de jogo e de conexão, assim como o meio para parar e iniciar o cronómetro de jogo.

Funcionalidades

O *Referee Box Command 2008* foi projectado para implementar as funcionalidades enumeradas nas tabelas 4.4 e 4.5, todavia este protótipo apenas implementa as funcionalidades enumeradas na tabela 4.4.

Apesar da aplicação não estar completa, o controlo e validação de eventos (fluxo de actividades da fig. 4.3) e a administração do sistema está numa fase muito próxima da final, mas a anulação e monitorização dos detalhes dos eventos e a retoma de jogos abortados, que consta da estrutura da aplicação, está nos primeiros passos de implementação.

[1] G.U.I - Graphical User Interface

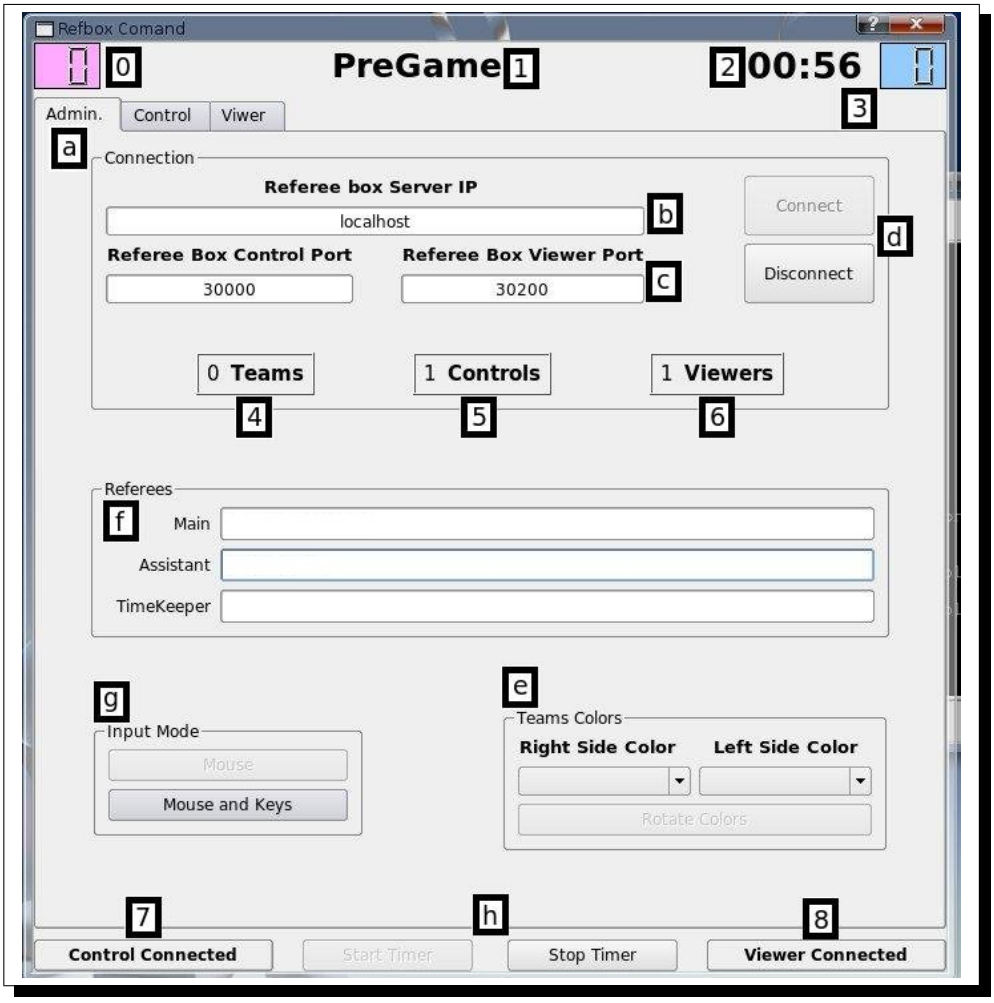


Figura 4.1: *Screen shot* do *frame* de administração (*Admin*) do *Referee Box Command 2008*

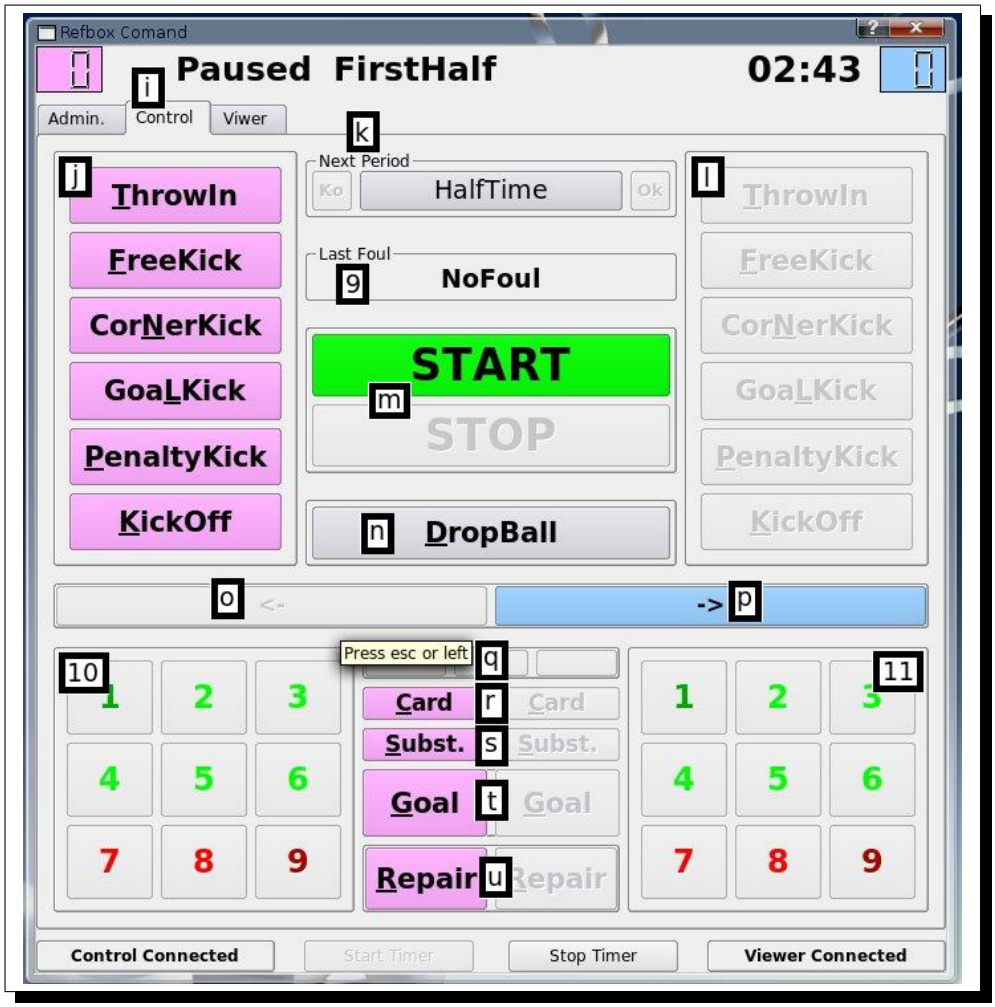


Figura 4.2: *Screen shot* do *frame* de controlo (*control*) do *Referee Box Command 2008*

Gerir e monitorizar a conexão com o servidor.	Frame *a* (admin), Linhas *b* e *c* e Botões *d*.
Identificar a equipa de arbitragem.	Frame *a* (admin), Linhas *f*.
Identificar a cor das equipas.	Frame *a* (admin), Rectângulo *e*.
Controlar o modo de introdução de eventos (Teclado ou Rato).	Frame *a* (admin), Rectângulo *g*.
Trocar (rodar) a disposição dos botões na interface de comando (ex. trocar botões Magenta da esq. para a dir. e os Cian da dir. para a esq.).	Frame *a* (control)
Controlar o cronómetro.	Botões *h*.
Introduzir eventos de jogo (Sempre que possível o sistema não permite a introdução de eventos incoerentes. - fig. 4.3).	Frame *i* (Control), os Botões dos Rectângulos *j, k l, m, q,r s, t, u, 10 e 11* e os Botões *n, o, p*.
Monitorizar cronómetro e tempo.	caixa, *2*.
Exibir Resultado.	caixas *0 e 3*.
Exibir período de jogo actual.	caixa *1*.
Exibir última falta.	caixa *i* (Control), Linha 9.
Exibir estado e tipo de jogadores registados.	Frame *i* (Control), Botões do Rectângulo 10 e 11.

Tabela 4.4: Funcionalidades implementadas no *Referee Box 2008 command*. As referências apontam para as figuras 4.1 e 4.2.

Consultar detalhes do jogo (Pormenores de jogadores e eventos como golos, faltas, cartões, substituições, reparações, fluxo de jogo e tempo).	Frame (Viewer).
Anular cartões, golos e mudanças de período de jogo.	Frame (Viewer).
Retomar um jogo abortado.	Frame (Admin).
Criar e terminar um jogo.	Frame (Admin).

Tabela 4.5: Funcionalidades não implementadas no *Referee Box 2008 command*. As referências apontam para as figuras 4.1 e 4.2.

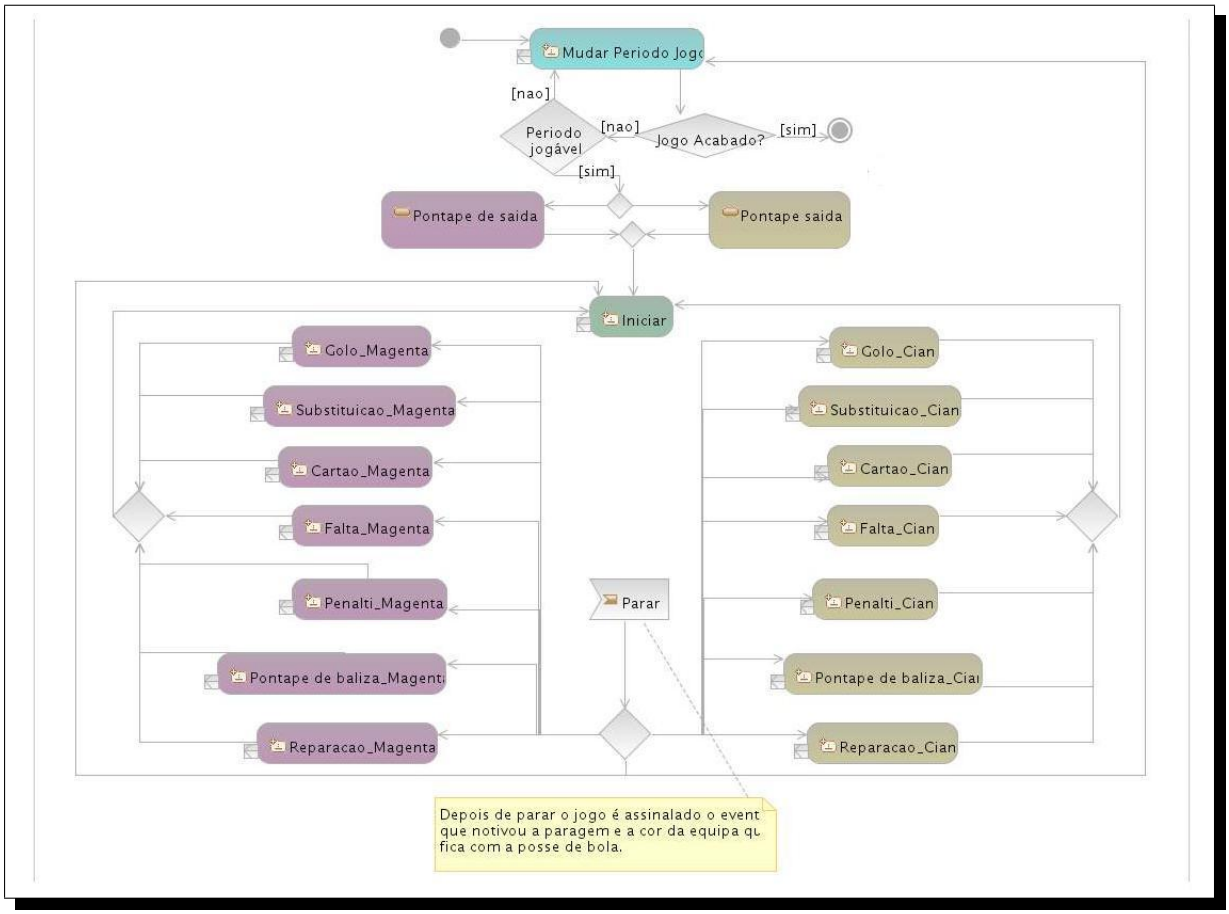


Figura 4.3: Fluxo de introdução de eventos de jogo validado pelo *Referee Box* command 2008. Apenas a sequência deste diagrama é permitida dentro do *frame* (control). Para o efeito os botões estão activos ou inactivos mediante o estado de jogo

Estado e Problemas

Este *Command 2008* revelou-se uma aplicação estável nas funcionalidades catalogadas como completas. Nesta configuração o sistema de rede, a introdução de eventos, as teclas de atalho, a "rotação" dos botões do frame de controlo, a activação de botões mediante o contexto e a exibição de informação, estão de acordo com os requisitos de projecto. Não obstante os inúmeros testes pelos quais a esta aplicação já passou considera-se que não estão esgotados. Mais concretamente as funcionalidades estáveis deverão ser testadas por pessoas não relacionadas com a sua construção e, como não pode deixar de ser, todo o sistema tem de ser novamente testado depois de completo.

O maior problema de estabilidade do *Command* é comum a todos os blocos da *Referee Box* e é gerado no parser. O protótipo do parser XML desenvolvido é funcional, mas é muito sensível a mensagens não conformes com o protocolo. Este facto faz com que uma falha na produção de mensagens XML numa qualquer aplicação se propague a todo o sistema.

4.2.2 *Referee Box* Proxy 2008

O *Referee Box* Proxy 2008 é uma aplicação escrita em C++/QT3 sobre Linux e implementa a retro compatibilidade de todo o sistema. Para o efeito serve de ponte entre o "novo" servidor e a equipa "antiga" e traduz as mensagens XML que fluem na rede da *Referee Box* 2008 para os caracteres do protocolo em vigor.

O *proxy* foi desenhado para fornecer aos árbitros ou às equipas uma ferramenta que permite que qualquer estação base actual funcione com a *Referee Box* 2008 sem nenhuma alteração.

Esta aplicação emula e regista uma equipa no sistema, criando para o efeito, uma conexão com o núcleo e enviando o conteúdo de um ficheiro de registo. Por outro lado, cria um porto de escuta na rede onde uma equipa "real" se conecta para receber as mensagens traduzidas.

Tanto as equipas como os árbitros podem instanciar esta aplicação. Se a *Referee Box* 2008 for adoptada, numa primeira fase, provavelmente serão os árbitros a instanciar e configurar o *proxy*. Numa segunda fase serão as equipas que não estiverem interessadas em mudar imediatamente de protocolo, numa terceira e última o *proxy* será eliminado. Isto acontecerá quando todas as equipas se adaptarem ao novo protocolo.

Interface com o Utilizador

O *Proxy* implementa duas interfaces. A primeira interface é de linha de comandos e apoia-se num menu de ajuda (tab. 4.6) onde são enumerados o conjunto de parâmetros que podem ser passados no lançamento. A segunda é gráfica (fig. 4.4) e permite a monitorização, o lançamento e o encerramento da conexão com o núcleo e do servidor do *proxy*, assim como a alteração dos parâmetros destes.

No modo gráfico não há a necessidade de relançar a aplicação em nenhuma operação o que não acontece no modo de linha de comandos. Este *proxy* pode se usado de maneira exclusivamente gráfica ou apenas na linha de comandos.

Option	Description	Default
-p port	Refbox Team Port	30100
-c port	Proxy Team Port	30300
-a adress	Refbox adress	localhost
-f file	Team Reg. File	team1.box
-g	GUI	
-i insert	Team Info.	
-h help	This menu	

Tabela 4.6: A lista de parâmetros pode ser consultada no menu de ajuda do *Referee Box Proxy 2008* (./RefboxProxy2008 -h)

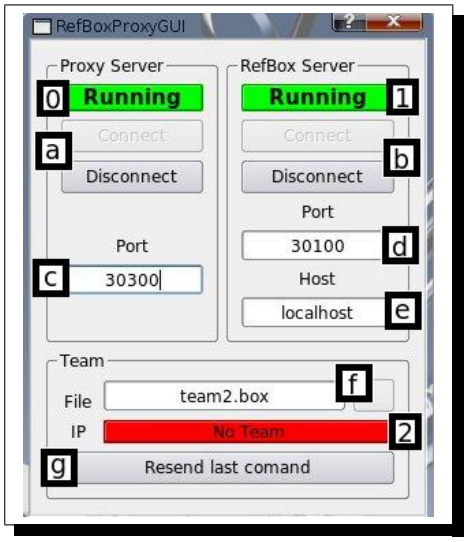


Figura 4.4: *Screen shot* da interface gráfica do *Referee Box Proxy 2008*

Funcionalidades

O *Referee Box Proxy 2008* foi projectado para implementar as funcionalidades enumeradas nas tabelas 4.7 e 4.8, todavia este protótipo apenas implementa as funcionalidades enumeradas na tabela 4.7.

Estado e Problemas

O *Referee Box proxy 2008* é uma aplicação estável nas funcionalidades catalogadas como completas. Esta aplicação, também foi exaustivamente testada. Todavia está incompleta e nunca foi testada por pessoas externas ao seu desenvolvimento.

Durante os testes em ambiente real, foi detectado e não corrigido um problema no sistema que permite que uma equipa receba todas as mensagens perdidas quando a conexão cai a meio de um jogo. Reparou-se que enviar uma sequência rápida de comandos pode ser nocivo para a integridade dos robôs, apesar de isso implicar que

Criar um porto de conexão para uma equipa.	Botão *a*.
Permitir que a conexão da equipa seja momentaneamente quebrada.	
Na re-conexão enviar à equipa mensagens perdidas.	Botão *g*.
Criar uma conexão com o núcleo da *Referee Box*.	Botão *b*.
Emular e registar uma equipa no núcleo (Enviar o ficheiro com os dados do registo).	Linha *f*.
Receber, traduzir e enviar mensagens do núcleo para a equipa.	
Se necessário, funcionar exclusivamente em modo não gráfico.	
Permitir o controlo e monitorização gráfica da conexão com o núcleo.	Caixa *1* e *2* Botões *b* Linha *d* e *e*.
Permitir o controlo e monitorização gráfica do servidor do proxy.	Caixa *0* Botões *a* Linha *c*.

Tabela 4.7: Funcionalidades implementadas do *Referee Box 2008 Proxy*. As referências apontam para a figura 4.4.

Navegar na árvore de ficheiros para seleccionar o ficheiro de registo da equipa.
Implementar uma interface para criar e gravar o ficheiro de registo.

Tabela 4.8: Funcionalidades por implementar do *Referee Box 2008 Proxy*.

eles transitem para o estado de jogo actual.

4.2.3 *Referee Box* Viewer 2008

O *Referee Box Viewer* é uma aplicação escrita em C++/QT3/OpenGL sobre Linux, que permite que o público e os elementos humanos das equipas visualizem o estado do jogo, assistam a animações sincronizadas com eventos de jogo e a animações publicitárias.

A função principal desta aplicação é exibir informações de jogo, nomeadamente o estado e tipo de equipas e jogadores em campo, cartões, golos, tempo, período de jogo, assim como animações destinadas ao público que assiste a uma partida.

Na *Referee Box* 2008, o *Viewer* pode ser multi-instanciado em qualquer nó da rede, ou seja, vários *Viewers* podem exibir simultaneamente a informação do mesmo jogo.

O *Referee Box Viewer* é um mero *front-end*, pelo que é desprovido de qualquer tipo de processamento relevante ao jogo. A acção deste sistema resume-se ao registo no núcleo e à exibição da informação recebida através dos relatórios XML que este lhe envia periodicamente.

Interface com o Utilizador

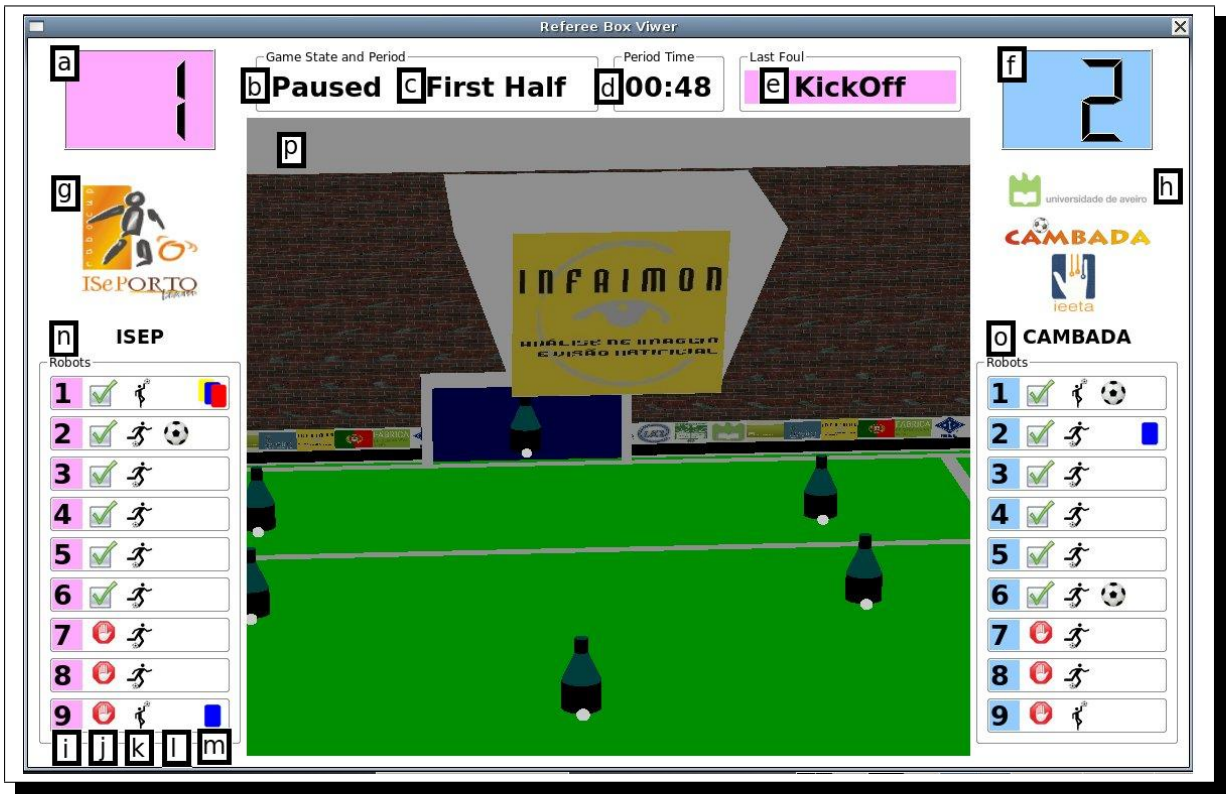


Figura 4.5: *Screen shot* da interface gráfica do *Referee Box Viewer 2008*

A interface gráfica do *Referee Box Viewer 2008* é apresentada na figura 4.5. A interface desta aplicação é dividida em duas partes: a primeira (rectângulo p), ocupa o centro, e consiste num *view port* "desenhado" (*rendered*) por um motor OpenGL; a

segunda, envolve a primeira, e consiste num conjunto de QT *widgets* [2] que exibem de forma gráfica os parâmetros mais importantes de jogo.

O *view port* OpenGL é uma área polivalente. Neste momento exibe, dentro de um estádio virtual, animações publicitárias sincronizadas com os eventos de jogo. É importante sublinhar que as animações apresentadas não são uma representação fiel do evento que aconteceu, são animações genéricas para cada evento, desenhadas para atrair a atenção do público para os painéis publicitários que revestem o estádio virtual 3D.

Apesar deste estádio não ter sido concebido com esse fim, se no futuro existir informação posicional dos robôs dentro da *Referee Box*, e mais concretamente dentro deste *Viewer*, poder-se-á representar a posição real dos jogadores e reconstruir jogadas.

Esta área central foi concebida também para no futuro poder apresentar texto com estatísticas de jogo, anúncios de partidas, avisos ao público, etc ....

Funcionalidades

Este *Viewer* foi desenhado para permitir que as pessoas presentes no "estádio" tenham mais informações sobre o jogo, assim como para criar uma via flexível e eficiente para exibir publicidade e informações de todo o tipo ao público.

O *Referee Box Viewer 2008* foi projectado para implementar as funcionalidades enumeradas nas tabelas 4.9 e 4.10. Todavia este protótipo apenas implementa as funcionalidades enumeradas na tabela 4.9.

Exibir Nome e logotipo da equipa.	Caixas *g,n,h,o.*
Exibir Resultado do jogo (Marcador).	Caixas *a* e *f.*
Exibir estado e período de jogo.	Caixas *b* e *c.*
Exibir tempo de jogo.	Caixa *e.*
Exibir robôs em jogo.	Caixas *n* e *o.*
Exibir o número de cada um dos robôs em jogo.	Caixa *i.*
Exibir estado de cada um dos robôs em jogo.	Caixa *j.*
Exibir tipo de cada um dos robôs em jogo.	Caixa *k.*
Exibir golos marcados por cada um dos robôs em jogo.	Caixa *l.*
Exibir cartões atribuídos a cada um dos robôs em jogo.	Caixa *m.*
Exibir animações publicitárias sincronizadas com eventos de jogo.	Caixa *p.*

Tabela 4.9: Funcionalidades implementadas do *Referee Box 2008 Viewer*. As referências apontam para a figura 4.5.

[2] Uma classe do Qt que impementa um componente gráfico denomina-se QT *widget*

Conexão ao endereço por defeito e re-conexão automática ao último endereço válido.	
Exibição de filmes, imagens, horários e informações estatísticas de jogo.	Caixa *p*
Controlo e configuração remota da informação não relacionada com o jogo (filmes, imagens e horários)	

Tabela 4.10: Funcionalidades não implementadas do *Referee Box 2008 Viewer* .

Estado e Problemas

O *Referee Box Viewer 2008*, nas funcionalidades catalogadas como completas, é uma aplicação funcional e testada. Todavia, à imagem das restantes aplicações da *Referee Box* 2008 o Viewer é um protótipo, pelo que há funcionalidades a acrescentar e mais testes a fazer.

Durante os testes em ambiente real os problemas identificados e não resolvidos têm a sua génese nas funcionalidades por implementar, nomeadamente a necessidade do *Viewer* se re-conectar automaticamente no caso de uma falha de conexão. Um aspecto a corrigir prende-se com o facto das animações programadas terem uma duração fixa, serem "desenhadas" em fila e serem agendadas em todos os eventos. Assim quando a cadência de eventos não é da ordem de grandeza da duração das animações há uma perca de sincronismo.

4.2.4 *Referee Box* Reporter 2008

Esta aplicação, escrita em C++ sobre Linux, lê o log produzido pelo núcleo (*Server Application*) e gera um documento (escrito em LateX). Este ficheiro TeX pode ser compilado num relatório de jogo legível (PDF ou num web site HTML).

Interface com o Utilizador

O interface do *Referee Box* Reporter 2008 é de linha de comandos e apoia-se-se num menu de ajuda (fig. 4.11) onde são enumerados o conjunto de parâmetros que podem ser passados no lançamento.

Nesta versão, para a criação do relatório em LateX, apenas é necessário apontar o nome e o caminho do ficheiro de entrada e o de saída.

Correr o *Reporter* não chega para produzir um ficheiro legível, o *Reporter* gera um ficheiro LateX que tem de ser compilado num relatório passível de ser lido e impresso, ou num web site. O compilador TeX também é uma aplicação de linha de comandos, pelo que, para agilizar o processo, os dois passos necessários para passar do ficheiro de *log* ao relatório podem feitos por um *script*.

Funcionalidades

O *Referee Box* Reporter gera um relatório com o formato apresentado no anexo D. Este relatório tem uma primeira página com o resumo das informações de jogo, uma

Option	Description
-f fName	XML msg from file
-o	.tex output file name
-h	This menu

Tabela 4.11: A lista de parâmetros pode ser consultada no menu de ajuda do *Referee Box Reporter 2008* (./RefboxLoger -h)

segunda com o resumo das informações das equipas registadas e uma terceira com o resumo das informações dos árbitros registados.

Estado e Problemas

A actual versão do *Referee Box Reporter 2008* está no estado final de implementação, pelo que foi pobremente testada, no entanto este protótipo já é capaz de gerar um relatório. Existem alguns campos do relatório (anexo D) que ainda não estão a ser gerados, nomeadamente os tempos; *(Start Time, Match Time, Game Time, Play-off Time e Net Time).*

Capítulo 5

Conclusões

Através do estudo da liga dos robôs médios e da sua *Referee Box*, feito no âmbito desta dissertação, conclui-se que para se obter uma arbitragem mais rigorosa, e para o público, equipas e árbitros terem acesso a mais e melhor informação, a *Referee Box* oficial teria de ser ampliada e reestruturada.

Através do estudo das tecnologias disponíveis para a construção de sistemas de comunicação e informação, e tendo em conta os recursos que existem no futebol robótico, conclui-se que a maneira mais simples, eficiente e modular de obter estas novas funcionalidades, passaria por distribuir a *Referee Box* na rede de campo, que actualmente já existe.

Tendo em conta a aplicação desenvolvida (*Referee Box* 2008), conclui-se que esta distribuição do sistema, sobre uma rede, é possível, permite a introdução de novas funcionalidades e acrescenta usabilidade ao sistema. Concluí-se também que esta abordagem apresenta vantagens no que diz respeito à capacidade de expansão e adaptatividade do sistema.

Com este trabalho demonstrou-se que este paradigma permite uma fácil integração de novas funcionalidades e aumenta a quantidade e a qualidade de informação que circula no sistema. Através do protocolo XML desenvolvido, as equipas podem produzir e ter acesso a relatórios detalhados, a partir dos quais, à posteriori ou em tempo real, podem extrair informação útil para o estudo táctico do jogo.

A *Referee Box* 2008, quando comparada com as restantes propostas, é uma solução mais escalável. A reutilização de componentes e a facilidade com que se pode fazer adições e alterações neste sistema, adequa-se este sistema *open sorce* e à multiplicidade de contribuições que pautam o desenvolvimento da *Referee Box*, neste meio.

Os mecanismos de retoma de conexões e jogos abortados, baseados na construção de um registo de jogo perene (ficheiro XML), projectados e parcialmente implementados no servidor da *Referee Box* 2008, já tornam este sistema mais fiável do que o sistema oficial da M.S.L.. Não obstante estes estarem incompletos, são viáveis e estão em avançado estado de execução. Quando estes mecanismos estiverem completamente operacionais, dotarão a *Referee Box* 2008, ou as suas sucessoras, de um conjunto de funcionalidades que aumentarão muito a tolerância do sistema a todo o tipo de falhas.

Não obstante as semelhanças com a interface da *Referee Box* oficial, a interface

da *Referee Box* 2008, nomeadamente as alterações que se baseiam na introdução de teclas de atalho, na ampliação do uso guiado e na introdução de um único painel de comando, provaram ser muito funcionais e apresentam-se como uma real mais valia para o sistema.

Apesar de pobremente implementada e testada, a introdução no projecto da *Referee Box* 2008 da possibilidade de "equipas mistas" [1] usarem o sistema, apresenta-se como uma funcionalidade útil e viável. Especialmente porque neste meio não é incomum as equipas não possuírem o número máximo de robôs que os regulamentos permitem, estando assim aberta a porta para duas ou mais equipas unirem esforços.

5.1 Teste do Sistema em Ambiente de Competição

O Robótica 2008, 8º Festival Nacional de Robótica, realizado em Aveiro de 2 a 6 de Abril foi a plataforma de testes em ambiente real da *Referee Box* 2008. A *Referee Box* 2008 foi o único sistema de apoio à arbitragem usado nos jogos e treinos de futebol robótico da liga dos robôs médios deste festival.

Neste festival, todas as aplicações, à excepção do *Referee Box* 2008 Reporter, foram usadas nas configurações apresentadas nesta dissertação. O *Referee Box* 2008 Reporter não foi usado porque, à data, estava ainda em desenvolvimento e não fornecia as funcionalidades necessárias ao seu uso. Durante os jogos desta competição, todas as aplicações da *Referee Box* 2008 utilizadas cumpriram as funções para que foram projectadas. O sistema funcionou apenas no modo de retro-compatibilidade, mas não houve nenhuma falha significativa nas funcionalidades catalogadas como estáveis.

Tendo em conta o teste descrito em cima, conclui-se que a *Referee Box* 2008, nomeadamente o lote de aplicações e funcionalidades já implementadas, são suficientes para apoiar a arbitragem de um jogo de futebol robótico em qualquer competição da M.S.L.. Conclui-se também que em áreas como a exibição de informação, adaptação à novas regras, multiplicidade e ergonomia das interfaces com o árbitro, a *Referee Box* 2008 já tem um desempenho superior à versão oficial do RoboCup.

Nesta competição, demonstrou-se que o sistema de exibição de animações publicitárias e informações de jogo, é uma via eficiente para esclarecer o público sobre alguns aspectos do jogo, bem como, para fazer com que este olhe para a publicidade do patrocinador da competição.

Neste ambiente tão experimental, poucas pessoas aceitam adicionar ao jogo o risco de uma falha da *Referee Box*. Por regra, as equipas, põem entraves à utilização de um sistema que desconhecem. As organizações das competições, em resultado do conservadorismo das equipas, só pode propor a utilização de um sistema que lhe dê uma garantia de fiabilidade consideravelmente superior ao sistema oficial.

Ao contrário dos sistemas das equipas, o sistema de arbitragem tem de ser usada por inúmeras pessoas, sem **nenhum** treino prévio e nas mais variadas condições. Este facto, faz com que a *Referee Box* tenha de ser um sistema muito fácil de usar e configurar.

[1] Uma equipa mista, é uma equipa constituída por várias equipas e respectivas estações base, que jogam cooperativamente contra um adversário comum.

Tendo em conta os dois dos requisitos básicos da *Referee Box* abordados em cima, a fiabilidade e a usabilidade, conclui-se que apesar da *Referee Box* 2008 já ser um sistema funcional, tem ainda algum caminho a percorrer para se impor neste meio.

No Robótica 2008, apercebemo-nos que para a *Referee Box* 2008 se transformar numa alternativa ao sistema oficial, é muito importante existir um sistema gráfico para o lançamento e configuração de todos as suas aplicações. Este sistema substituirá os *scripts* que foram usados e servirá para minorar a complexidade gerada pela necessidade de lançar e configurar uma ou mais instâncias de cinco aplicações diferentes. Para o efeito, poderá ser criado um wizard [2] que o(s) utilizador(s) do sistema corre(m) em cada um dos nós para instanciar a *Referee Box* 2008 na configuração que pretende(m).

No Robótica 2008 não foi possível comunicar directamente com nenhuma equipa através do protocolo XML, pelo que algumas mais valias da *Referee Box* 2008 não puderam ser exploradas pelas equipas. Neste cenário o sistema que permite a compatibilidade da *Referee Box* 2008 com equipas que usam o protocolo oficial, foi de uma utilidade extrema. Pelos motivos citados em cima é muito complexo convencer as equipas a fazer extensas adaptações nas suas estações bases, a menos que lhes seja dado muito tempo e estas vejam mais valias óbvias na mudança.

Durante o Robótica 2008, confirmaram-se as dificuldades que um único operador tem para introduzir correctamente toda a informação no sistema. A *Referee Box* 2008 cumpre as novas regras, que obrigam ao controlo das substituições e reparações. Todavia, concluiu-se que não é aceitável transferir também essa responsabilidade para o já sobrecarregado operador do sistema. Este facto vem sustentar, ainda mais, o multi-controlo e a multi-monitorização, permitida e parcialmente implementada na *Referee Box* 2008.

5.2 Trabalho futuro

No que diz respeito ao árbitro e às equipas, a configuração actual e as funcionalidades da *Referee Box* 2008, não são muito diferentes das da *Referee Box* oficial. Todavia o paradigma e o potencial são muito diferentes. Nesse sentido, para trabalho futuro, a prioridade deve ser a conclusão das funcionalidades projectadas, nomeadamente a retoma de jogos e conexões abortadas, o cálculo e exibição de estatísticas de jogo. As funcionalidades por implementar nas aplicações da *Referee Box* 2008 foram enumeradas no capítulo 4 nas tabelas 4.3, 4.5, 4.8 e 4.10.

No futuro, é também importante testar mais o sistema e resolver os problemas identificados ou a identificar.

Uma das motivações iniciais para a construção deste sistema, foi a construção de interfaces portáteis dedicadas a cada árbitro. Esse objectivo ainda não foi cumprido, porque primeiro, quis-se provar que este sistema poderia ter o aspecto e operar da mesma forma que o sistema oficial. Ao mesmo tempo que se implementou uma interface parecida com a que actualmente é usada, que já foi testada e aceite, deixou-se a porta

[2] Um wizard é uma interface gráfica "amiga do utilizador", onde é exibida uma sequência de caixas, para executar uma sequência de passos no sentido de cumprir uma determinada acção.

aberta para a implementação de várias interfaces que nada têm a ver com o sistema actual, que são o corolário de todo este sistema.

Em simultâneo com o desenvolvimento e teste desta *Referee Box*, foram propostas várias vias e sistemas para implementar interfaces individuais para os árbitros, todavia nenhuma delas passou à prática. Os sistemas equacionados são enumerados a seguir:

- Construção de uma ou várias aplicações de commando que corressem em PDAs ou dispositivos portáteis. Estes dispositivos usariam uma rede sem fios para se ligarem ao *Referee Box* server e funcionariam de maneira igual ao actual *Referee Box* 2008 Command.

- Construção de um ou vários comandos, com um pequeno visor e com teclas reais, dedicado a controlar as funções mais importantes de cada um dos árbitros em jogo. Este *hardware* enviaria por rádio as suas ordens para uma base, que as introduziria na rede de campo.

- Construção de um aplicação, que através do uso de um sistema áudio sem fios e da rede de campo, receberia e traduziria as ordens verbais dos árbitros, inserindo-as de seguida na rede de campo. Esta aplicação, poderia também descrever aos árbitros o impacto das ordens dadas, pela mesma via áudio.

Bibliografia

[1] http://world.honda.com/ASIMO/, Junho 2008. Sítio na Internet do robô humanóide ASIMO - Advanced Step in innovative Mobility.

[2] http://www.ieeta.pt/atri, Maio 2008. Sítio na Internet do ATRI - Actividade Transversal em Robótica Inteligente.

[3] http://www.ieeta.pt/atri/cambada/, Junho 2007. Sítio na Internet do CAMBADA - Cooperative Autonomous Mobile roBots with Advanced Distributed Architecture.

[4] RoboCup Technical Committee. Robocup2007 poster. http://www.er.ams.eng.osaka-u.ac.jp/robocup-mid/index.cgi?action=ATTACH&page=Rules+and+Regulations&file=PosterRoboCup2007%2Epdf, 2007.

[5] RoboCup Technical Committee. *Laws of the F180 League 2008*. RoboCup, http://small-size.informatik.uni-bremen.de/_media/rules:ssl-rules-2008.pdf?id=rules%3Amain&cache=cache, 2008. Regras da liga dos robôs pequenos do RoboCup.

[6] RoboCup Technical Committee. *RoboCup Four-Legged League Rule Book*. RoboCup, http://www.tzi.de/4legged/pub/Website/Downloads/AiboRules2008.pdf, Janeiro 2008. Regras da liga dos robôs com quatro pernas do RoboCup.

[7] RoboCup Technical Committee. *RoboCup Standard Platform Rule Book*. RoboCup, http://www.tzi.de/4legged/pub/Website/Downloads/NaoRules2008.pdf, Fevereiro 2008. Regras da liga de plataforma comum.

[8] Gustavo A Corrente. Robocup middle size league referee box. *IADIS - APPLIED COMPUTING 2007*, Fevereiro 2007.

[9] http://www.fifa.com/, Junho 2008. Sítio na Internet da FIFA - Fédération Internationale de Football Association.

[10] M. Fowler and K. Scott. *UML Distilled - Applying the Standard Object Modeling Language*. Addison-Wesely, 1997.

[11] http://www.fsf.org/, Dezembro 2008. Sítio na Internet da Free Software Fondation - promotora da G.P.L.

[12] http://www.ieeta.pt/, Junho 2007. Sítio na Internet do IEETA - Instituto de Engenharia Electrónica e Telemática da Universidade de Aveiro.

[13] Marques J. and Guedes P., editors. *Tecnologia de Sistemas Distribuídos*. FCA, 1998.

[14] http://sserver.sourceforge.net/wiki/index.php/Main_Page, Junho 2008. Sítio na Internet do Liga RoboCup de Simulação.

[15] http://www.humanoidsoccer.org/, Junho 2008. Sítio na Internet da Liga RoboCup dos robôs humanóides.

[16] http://www.er.ams.eng.osaka-u.ac.jp/robocup-mid/index.cgi, Junho 2007. Sítio na internet da Liga RoboCup dos Robôs Médios.

[17] http://small-size.informatik.uni-bremen.de/, Junho 2008. Sítio na Internet da Liga RoboCup dos robôs pequenos.

[18] Asada M., Balch T., Bonarini A., and et al. *Middle Size Robot League Rules and Regulations for 2007*. RoboCup, http://www.er.ams.eng.osaka-u.ac.jp/robocup-mid/index.cgi?action=ATTACH&page=Rules+and+Regulations&file=msl%2Drules%2D2007%2D11%2D09%2Epdf, 10.5 edition, Janeiro 2007. Regras da liga dos robôs médios do RoboCup.

[19] http://sourceforge.net/projects/msl-refbox, Junho 2007. Sítio na Internet da MSL Referee Box.

[20] I. Oliveira and J. P. Cunha. *SLiM - SIAS Light Method*. DETIUA - IEETA - SIAS, 2002.

[21] http://www.opengl.org/, Junho 2007. Sítio na Internet OpenGL.

[22] http://www.robocup.org, Junho 2008. Sítio na Internet do RoboCup.

[23] Behnke S. *RoboCupSoccer Humanoid League Rules and Setup for the 2006 competition in Bremen, Germany*. RoboCup, http://www.humanoidsoccer.org/rules/HumanoidLeagueRules2006.pdf, 7 edition, Junho 2006. Regras da liga dos robôs humanóides do RoboCup.

[24] Mullender S., editor. *Distributed Systems*. Frontier Series, 1990.

[25] Mullender S., editor. *Distributed Systems - (Second Edition)*. Addison-Wesley, 1994.

[26] http://www.ieeta.pt/SIAS/, Dezembro 2007. Sítio na Internet do SIAS - Grupo de Sistemas de Informação na Área da Saúde do IEETA - Instituto de Engenharia Electrónica e Telemática da Universidade de Aveiro.

[27] http://www.ua.pt, Junho 2007. Sítio na Internet da UA - Universidade de Aveiro.

Anexos

Apêndice A

Mensagens Codificadas em XML

O protocolo XML da *Referee Box* 2008 foi desenhado para servir de registo e para servir de meio de transferência de informação entre os vários módulos do sistema.

A função de registo consiste na escrita sequencial das mensagens num ficheiro dentro do *Referee Box* 2008 Server.

A versão completa das mensagens "Complete/Server/Log version" é gerada exclusivamente dentro do núcleo, com fim de registo ou envio.

As aplicações clientes têm de ter capacidade de ler as versões completas das mensagens, mas apenas necessitam de gerar e emitir versões simplificadas destas "Control Sender version". No entanto qualquer entidade, se assim o entender, pode gerar mensagens completas. Todavia o núcleo, quando recebe uma mensagem, limita-se a integrar a informação obrigatória desta. A restante informação é calculada, de seguida a nova mensagem é guardada e, se for necessário, é reencaminhada.

A.1 Action

```
<!-- Action message -->
<!-- this message is used to start and stop the game -->

<!-- ############################################################################ -->
<!-- Complete/Server/Log version -->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/> <!--optional tag to sender -->
        <ControlEvent>
                <GameAction state="Start" period="FirstHalf" periodTime="31"
                        eventNumber="009001"/>
                <!--eventNumber, period and periodTime atributes are optional
                        to sender -->
        </ControlEvent>
</RefereeBoxProtocol>

<!-- ############################################################################ -->
<!-- Control Sender version -->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <GameAction state="Start"/>
        </ControlEvent>
</RefereeBoxProtocol>

<!-- ############################################################################ -->
```

```
<!-- Options
        state="Start"
        state="Stop"

        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"
-->
```

A.2 Admin

```
<!-- Game administration message -->
<!-- this message is used to administrate referee box server-->

<!-##########################################################################->
<!-- Complete/Server/Log version-->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>     <!--optional tag to sender-->
        <ControlEvent>
                <GameAdmin order="kill" gameNumber="00001" gameLocation="UA"
                        period="FirstHalf" periodTime="31"/>
                <!--if order="kill" all other atributes are optional-->
                <!--if order="resume" gameNumber is mandatory to sender and all
                        other atributes are optional -->
                <!--if order="launch" gameNumber and gameLocation is mandatory
                        to sender, period and periodTime ate optionals-->
        </ControlEvent>
</RefereeBoxProtocol>

<!-##########################################################################->
<!-- Control Sender version-->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <GameAdmin order="kill"/>
        </ControlEvent>
</RefereeBoxProtocol>

<!--or-->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <GameAdmin order="resume" gameNumber="00001" />
        </ControlEvent>
</RefereeBoxProtocol>

<!--or-->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <GameAdmin order="launch" gameNumber="00001" gameLocation="UA"/>
        </ControlEvent>
</RefereeBoxProtocol>

<!-##########################################################################->
<!-- Options
        order="launch"          Start new Game
        order="kill"            end actual Game
        order="resume"          resume one gamne data
 -->
```

A.3 Card

```
<!-- Card message-->
<!-- this message is used to send or anull one player card-->

<!-##########################################################################->
<!-- Complete/Server/Log version-->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>     <!--optional tag to sender-->
        <ControlEvent>
                <Card color="yellow" period="FirstHalf" periodTime="31"
                        eventNumber="1" option="Remove">
                        <!--period and periodTime atributes are allways optional-->
                        <!--eventNumber is optional to sender if option="add" -->
                        <!--color is optional to sender if option="Remove" -->
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                                <!--role and state atributes are optional -->
                </Card>
        </ControlEvent>
</RefereeBoxProtocol>

<!-##########################################################################->
<!-- Control Sender version-->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <Card color="yellow" option="add">
                        <Player color="Magenta" number="1"/>
                </Card>
        </ControlEvent>
</RefereeBoxProtocol>

<!--or-->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <Card eventNumber="005004" option="Remove">
                        <Player color="Magenta" number="1"/>
                </Card>
        </ControlEvent>
</RefereeBoxProtocol>

<!-##########################################################################->
<!-- Options
        <Card color="Yellow"
        <Card color="Blue"
        <Card color="Red"

        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"

        option="Remove"
        option="add"

        role="Goalkeeper" | "GK"
        role="FieldPlayer" | "FP"

        state="PlayerOut" | "OUT"
        state="PlayerIn" | "IN"
        state="PlayerSuspended" | "SU"
        state="PlayerSentOff" | "SO"

        <Player color="Cyan"
        <Player color="Magenta"
        <Player color="Yellow"
```

```
        <Player color="Blue"
        <Player color="Red"
        <Player color="Green"
-->
```

A.4 Foul

```
<!-- Foul message -->
<!-- this message is used to set next game state or to indicate previous
foul event-->

<!-############################################################################->
<!-- Complete/Server/Log version-->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>      <!--optional tag to sender-->
        <ControlEvent>
                <GameFoul foul="FreeKick" color="Cyan" period="FirstHalf"
                        periodTime="31" eventNumber="1"/>
                        <!--eventNumber, period and periodTime atributes are
                                optional to sender-->
        </ControlEvent>
</RefereeBoxProtocol>

<!-############################################################################->

<!-- Control Sender version-->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <GameFoul foul="FreeKick" color="Cyan"/>
        </ControlEvent>
</RefereeBoxProtocol>

<!--or-->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <GameFoul foul="DropBall"/>
        </ControlEvent>
</RefereeBoxProtocol>

<!-############################################################################->
<!-- Options
        foul="KickOff"
        foul="PenaltyKick"
        foul="ThrowIn"
        foul="CornerKick"
        foul="GoalKick"
        foul="DropBall"

        color="Cyan"
        color="Magenta"
        color="Yellow"
        color="Blue"
        color="Red"
        color="Green"

        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"
-->
```

A.5 Goal

```
<!-- Goal message -->
<!-- this message is used to send or anull one goal-->

<!-##########################################################################->
<!-- Complete/Server/Log version-->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>     <!--optional tag to sender-->
        <ControlEvent>
                <Goal color="Magenta" period="FirstHalf" periodTime="31"
                        eventNumber="1" option="Remove">
                        <!--period and periodTime atributes are optional -->
                        <!--eventNumber is optional to sender if option="add"-->
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                                <!--role and state atributes are optional -->
                </Goal>
        </ControlEvent>
</RefereeBoxProtocol>


<!-##########################################################################->
<!-- Control Sender version-->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <Goal color="Magenta" option="add">
                        <Player color="Magenta" number="1"/>
                <Goal/>
        </ControlEvent>
</RefereeBoxProtocol>

<!--or-->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <Goal color="Magenta" eventNumber="005004" option="Remove">
                        <Player color="Magenta" number="1"/>
                </Goal>
        </ControlEvent>
</RefereeBoxProtocol>

<!-##########################################################################->
<!-- Options
        color="Cyan"
        color="Magenta"
        color="Yellow"
        color="Blue"
        color="Red"
        color="Green"

        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"

        option="Remove"
        option="add"

        role="Goalkeeper" | "GK"
        role="FieldPlayer" | "FP"

        state="PlayerOut" | "OUT"
        state="PlayerIn" | "IN"
        state="PlayerSuspended" | "SU"
        state="PlayerSentOff" | "SO"
-->
```

A.6 Control Registration

```
<!-- Control connect/Register Message -->
<!-- this message is used to Register one Control system and his controlers
in Refreebox server -->

<!-- ########################################################################## -->
<!-- Complete/Server/Log version -->

<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>     <!--optional tag to sender-->
        <ControlRegistration type="GlobalControl" period="FirstHalf"
                periodTime="31">
                <!-- all ControlRegistration atributes are optionals-->
                <Controlers>
                        <!--REFEREE LIST MIN=0 - MAX=3 -->
                        <Referee type="MainReferee" name="Gustavo"
                                email="xpto@ua.pt"/>
                        <Referee type="AssisReferee" name="Sergio"
                                email="xpto@ua.pt"/>
                        <Referee type="TimeKeeper" name="Artur"
                                email="xpto@ua.pt"/>
                </Controlers>
        </ControlRegistration>
</RefereeBoxProtocol>

<!--or-->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>     <!--optional tag to sender-->
        <ControlRegistration type="GlobalControl" period="FirstHalf"
                periodTime="31"/>
                <!-- all ControlRegistration atributes are optionals to sender -->
</RefereeBoxProtocol>

<!-- ########################################################################## -->
<!-- Control Sender version -->

<RefereeBoxProtocol version="2.0">
        <ControlRegistration type="GlobalControl">
                <Controlers>
                        <!--REFEREE LIST MIN=0 - MAX=3 -->
                        <Referee type="MainReferee" name="Gustavo"
                                email="xpto@ua.pt"/>
                        <Referee type="AssisReferee" name="Sergio"
                                email="xpto@ua.pt"/>
                        <Referee type="TimeKeeper" name="Artur"
                                email="xpto@ua.pt"/>
                </Controlers>
        </ControlRegistration>
</RefereeBoxProtocol>

<!--or-->
<RefereeBoxProtocol version="2.0">
        <ControlRegistration type="GlobalControl"/>
</RefereeBoxProtocol>

<!-- ########################################################################## -->
<!-- Options
        period="PreGame" || "PG"
        period="FirstHalf" || "FH"
        period="HalfTime" || "HT"
        period="SecondHalf" || "SH"
        period="PenaltyShootOut" || "PS"
        period="EndGame" || "EG"

        <ControlRegistration type="GlobalControl" || "GC"
```

```
        <ControlRegistration type="StatStopGoalFoulCardControl" || "SSGFCC"
        <ControlRegistration type="StartStopControl" || "SSC"
        <ControlRegistration type="InOutControl" || "IOC"
        <ControlRegistration type="GoalControl" || "GC"
        <ControlRegistration type="FoulControl" || "FC"
        <ControlRegistration type="CardControl" || "CC"

        <Referee type="MainReferee" || "MR"
        <Referee type="AssisRefree" || "AR"
        <Referee type="TimeKeeper" || "TK"
-->
```

A.7 Team Registration

```
<!-- Team connect/Register message-->
<!-- this message is used to Register one Base station, his teams and
players in Refreebox server-->
<!-##########################################################################->
<!-- Complete/Server/Log version-->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>      <!--optional tag to sender-->
        <TeamRegistration type="DirectTeam" period="FirstHalf"
                periodTime="31" >
                <!-- all TeamRegistration atributes are optionals to sender -->
                <Team color="Magenta" name="CAMBADA" leader="JLA"
                        email="xpto@ua.pt">
                        <!--only color and name atributes are mandatory to sender-->
                        <!--PLAYER LIST MIN=1 - MAX=9 -->
                        <Player color="Magenta" number="1" role="GK"
                                state="IN"/>
                        <Player color="Magenta" number="2" role="FP"
                                state="PlayerIn"/>
                        <Player color="Magenta" number="3" role="FP"
                                state="IN"/>
                        <Player color="Magenta" number="4" role="FP"
                                state="IN"/>
                        <Player color="Magenta" number="5" role="FieldPlayer"
                                state="IN"/>
                        <Player color="Magenta" number="6" role="FP"
                                state="IN"/>
                        <Player color="Magenta" number="7" role="FP"
                                state="OUT"/>
                        <Player color="Magenta" number="8" role="FieldPlayer"
                                state="OUT"/>
                        <Player color="Magenta" number="9" role="Goalkeeper"
                                state="PlayerOut"/>
                                <!--color atribute is optional tag to sender-->
                </Team>
        </TeamRegistration>
</RefereeBoxProtocol>

<!-##########################################################################->
<!-- Team Sender version-->
<RefereeBoxProtocol version="2.0">
        <TeamRegistration type="DirectTeam">
                <Team color="Magenta" name="CAMBADA" leader="JLA"
                        email="xpto@ua.pt">
                        <!--only color and name atributes are mandatory-->
                        <!--PLAYER LIST MIN=1 - MAX=9 -->
                        <Player number="1" role="GK" state="IN"/>
                        <Player number="2" role="FP" state="PlayerIn"/>
                        <Player number="3" role="FP" state="IN"/>
                        <Player number="4" role="FP" state="IN"/>
                        <Player number="5" role="FieldPlayer" state="IN"/>
                        <Player number="6" role="FP" state="IN"/>
                        <Player number="7" role="FP" state="OUT"/>
                        <Player number="8" role="FieldPlayer" state="OUT"/>
                        <Player number="9" role="Goalkeeper" state="PlayerOut"/>
                </Team>
        </TeamRegistration>
</RefereeBoxProtocol>

<!-##########################################################################->
<!-- Options
        type="DirectTeam"
        type="ProxyTeam"
```

```
        color="Cyan"
        color="Magenta"
        color="Yellow"
        color="Blue"
        color="Red"
        color="Green"

        role="Goalkeeper" | "GK"
        role="FieldPlayer" | "FP"

        state="PlayerOut" | "OUT"
        state="PlayerIn" | "IN"
        state="PlayerSuspended" | "SU"
        state="PlayerSentOff" | "SO"
-->
```

A.8 Viewer Registration

```
<!-- Viwere connect/Register message -->
<!-- this message is used to Register one Viwere system in
Refree box server-->

<!-############################################################################->
<!-- Complete/Server/Log version-->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>      <!--optional tag->
        <ViewerRegistration type="PublicViewer" period="FirstHalf"
                periodTime="31" />
                <!-- all TeamRegistration atributes are optionals-->
</RefereeBoxProtocol>

<!-############################################################################->
<!-- Viewer Sender version-->
<RefereeBoxProtocol version="2.0">
        <ViewerRegistration type="PublicViewer"/>
</RefereeBoxProtocol>

<!-- Options
        type="TeamViewer"
        type="RefereeViewer"
        type="PublicViewer"

        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"
-->
```

A.9 Repair

```
<!-- Repair Message -->
<!-- this message is used to Register and comunicate players Entries and
Exits to repair purposes -->
<!-##########################################################################->
<!-- Complete/Server/Log version -->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>      <!--optional tag to sender-->
        <ControlEvent>
                <Repair period="PreGame" periodTime="31" eventNumber="1">
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                                <!--role and color atributes are optional-->
                </Repair>
                <!--period and periodTime atributes are optional to sender-->
        </ControlEvent>
</RefereeBoxProtocol>

<!-##########################################################################->
<!-- Control Sender version -->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <Repair>
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                </Repair>
        </ControlEvent>
</RefereeBoxProtocol>

<!-##########################################################################->
<!-- Options
        color="Cyan"
        color="Magenta"
        color="Yellow"
        color="Blue"
        color="Red"
        color="Green"

        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"

        role="Goalkeeper" | "GK"
        role="FieldPlayer" | "FP"

        state="PlayerOut" | "OUT"
        state="PlayerIn" | "IN"
        state="PlayerSuspended" | "SU"
        state="PlayerSentOff" | "SO"
-->
```

A.10 Match Report

```
<!-- Match Report -->
<!-- this message describe the game at the momment. it is emited eatch
second to every registed Viweres systems -->
<!-##########################################################################->
<!-- Complete/Server/Log version -->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>      <!--optional tag to sender-->
        <MatchReport>
                <Timer period="PreGame" totalPeriodTime="1" totalGameTime="2"
                        playedPeriodTime="3" playedGameTime="4"  running="true"/>

                <GameAdmin order="launch" gameNumber="00001" gameLocation="UA"
                        period="FirstHalf" periodTime="31"/>

                <!--TEAMS LIST MIN=0 - MAX=18 -->
                <Team color="Magenta" name="CAMBADA" leader="JLA" email="xpto@ua.pt">
                        <!--PLAYER LIST MIN=1 - MAX=9 -->
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                        <Player color="Magenta" number="2" role="FP" state="IN"/>
                        <Player color="Magenta" number="3" role="FP" state="IN"/>
                        <Player color="Magenta" number="4" role="FP" state="IN"/>
                        <Player color="Magenta" number="5" role="FP" state="IN"/>
                        <Player color="Magenta" number="6" role="FP" state="IN"/>
                        <Player color="Magenta" number="7" role="FP" state="OUT"/>
                        <Player color="Magenta" number="8" role="FP" state="OUT"/>
                        <Player color="Magenta" number="9" role="GK" state="OUT"/>
                </Team>


                <!--CARD LIST MIN=0 - MAX=60 -->
                <Card color="yellow" period="FirstHalf" periodTime="31"
                        eventNumber="1" option="Remove">
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                <Card/>


                <!--GOAL LIST MIN=0 - MAX=100 -->
                <Goal color="Magenta" period="FirstHalf" periodTime="31"
                        eventNumber="1" option="Remove">
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                <Goal/>


                <!--FOUL LIST MIN=0 - MAX=150 -->
                <GameFoul foul="FreeKick" color="Cyan" period="FirstHalf"
                        periodTime="31" eventNumber="1"/>


                <!--SUBST LIST MIN=0 - MAX=100 -->
                <Substitution period="PreGame" periodTime="31" eventNumber="1">
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                        <Player color="Magenta" number="2" role="FP" state="OUT"/>
                </Substitution>


                <!--REPAIRS LIST MIN=0 - MAX=100 -->
                <Repair period="PreGame" periodTime="31" eventNumber="1">
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                </Repair>

                <!--Last Action-->
                <GameAction state="Start" period="FirstHalf" periodTime="31"
                        eventNumber="1"/>
```

```
        </MatchReport>
</RefereeBoxProtocol>

<!-- ############################################################################## -->
<!-- Options
        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"

        running="true"
        running="false"

        <!-- Card color tag -->
        <Card color="Yellow"
        <Card color="Blue"
        <Card color="Red"

        <!-- All other color tags -->
        color="Cyan"
        color="Magenta"
        color="Yellow"
        color="Blue"
        color="Red"
        color="Green"

        role="Goalkeeper" | "GK"
        role="FieldPlayer" | "FP"

        state="PlayerOut" | "OUT"
        state="PlayerIn" | "IN"
        state="PlayerSuspended" | "SU"
        state="PlayerSentOff" | "SO"

        state="Start"
        state="Stop"

        foul="KickOff"
        foul="PenaltyKick"
        foul="ThrowIn"
        foul="CornerKick"
        foul="GoalKick"
        foul="DropBall"

-->
```

A.11 Server Report

```
<!-- Server Report -->
<!-- this message describe the system arquitecture at the momment. it
is emited eatch second to every registed Controls systems -->
<!-##########################################################################->
<!-- Complete/Server/Log version -->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>      <!--optional tag to sender-->
        <ServerReport>
                <Timer period="PreGame" totalPeriodTime="1" totalGameTime="2"
                        playedPeriodTime="3" playedGameTime="4"  running="true"/>

                <GameAdmin order="launch" gameNumber="00001" gameLocation="UA"
                        period="FirstHalf" periodTime="31"/>

                <!-- TEAM LIST MIN=0 - MAX=18 -->
                <TeamRegistration type="DirectTeam" period="SecondHalf"
                        periodTime="31">
                        <Team color="Magenta" name="CAMBADA" leader="JLA"
                                email="xpto@ua.pt">
                                <!--PLAYER LIST MIN=1 - MAX=9 -->
                                <Player color="Magenta" number="1" role="GK"
                                        state="IN"/>
                                <Player color="Magenta" number="2" role="FP"
                                        state="IN"/>
                                <Player color="Magenta" number="3" role="FP"
                                        state="IN"/>
                                <Player color="Magenta" number="4" role="FP"
                                        state="IN"/>
                                <Player color="Magenta" number="5" role="FP"
                                        state="IN"/>
                                <Player color="Magenta" number="6" role="FP"
                                        state="IN"/>
                                <Player color="Magenta" number="7" role="FP"
                                        state="OUT"/>
                                <Player color="Magenta" number="8" role="FP"
                                        state="OUT"/>
                                <Player color="Magenta" number="9" role="GK"
                                        state="OUT"/>
                        </Team>
                </TeamRegistration>


                <!-- CONTROLS LIST MIN=0 - MAX=50 -->
                <ControlRegistration type="GlobalControl" periodTime="31"
                        periodTime="12">
                        <Controlers>
                                <!--REFEREE LIST MIN=0 - MAX=3 -->
                                <Referee type="MainReferee" name="Gustavo"
                                        email="xpto@ua.pt"/>
                                <Referee type="AssisReferee" name="Sergio"
                                        email="xpto@ua.pt"/>
                                <Referee type="TimeKeeper" name="Artur"
                                        email="xpto@ua.pt"/>
                        </Controlers>
                </ControlRegistration>


                <!-- VIEWERS LIST MIN=0 - MAX=50 -->
                <ViewerRegistration type="PublicViewer" period="EndGame"
                        periodTime="31"/>

        </ServerReport>
</RefereeBoxProtocol>
```

```
<!-- ########################################################################## -->
<!-- Simplified/Server/Log version -->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0" />      <!-- optional tag to sender -->
        <ServerReport>
                <Timer period="PreGame" totalPeriodTime="1" totalGameTime="2"
                        playedPeriodTime="3" playedGameTime="4"
                                running="true" />

                <GameAdmin order="launch" gameNumber="00001" gameLocation="UA"
                        period="FirstHalf" periodTime="31" />

                <!-- TEAM LIST MIN=0 - MAX=18 -->
                <TeamRegistration type="DirectTeam" period="SecondHalf"
                        periodTime="31" />


                <!-- CONTROLS LIST MIN=0 - MAX=50 -->
                <ControlRegistration type="GlobalControl" period="SecondHalf"
                        periodTime="31" />


                <!-- VIEWERS LIST MIN=0 - MAX=50 -->
                <ViewerRegistration type="PublicViewer" period="EndGame"
                        periodTime="31" />
        </ServerReport>
</RefereeBoxProtocol>

<!-- ########################################################################## -->
<!-- Options
        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"

        running="true"
        running="false"

        <TeamRegistration type="DirectTeam"
        <TeamRegistration type="ProxyTeam"

        color="Cyan"
        color="Magenta"
        color="Yellow"
        color="Blue"
        color="Red"
        color="Green"

        role="Goalkeeper" | "GK"
        role="FieldPlayer" | "FP"

        state="PlayerOut" | "OUT"
        state="PlayerIn" | "IN"
        state="PlayerSuspended" | "SU"
        state="PlayerSentOff" | "SO"


        <ControlRegistration type="GlobalControl" || "GC"
        <ControlRegistration type="StatStopGoalFoulCardControl" || "SSGFCC"
        <ControlRegistration type="StartStopControl" || "SSC"
        <ControlRegistration type="InOutControl" || "IOC"
        <ControlRegistration type="GoalControl" || "GC"
        <ControlRegistration type="FoulControl" || "FC"
        <ControlRegistration type="CardControl" || "CC"
```

```
        <Referee type="MainReferee" || "MR"
        <Referee type="AssisRefree" || "AR"
        <Referee type="TimeKeeper" || "TK"

        <ViewerRegistration type="TeamViewer"
        <ViewerRegistration type="RefereeViewer"
        <ViewerRegistration type="PublicViewer"
-->
```

A.12 Substitution

```
<!-- Substitution Message -->
<!-- this message is used to Register and comunicate players Substitutions -->

<!-############################################################################-->
<!-- Complete/Server/Log version -->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>     <!--optional tag to sender-->
        <ControlEvent>
                <Substitution period="PreGame" periodTime="31" eventNumber="1">
                        <Player color="Magenta" number="1" role="GK" state="IN"/>
                        <Player color="Magenta" number="2" role="FP" state="OUT"/>
                                <!--role and color atributes are optional-->
                                <!--Player atributes represent the new
                                        player state -->
                </Substitution>
                <!--period and periodTime atributes are optional to sender-->
        </ControlEvent>
</RefereeBoxProtocol>

<!-############################################################################-->
<!-- Control Sender version -->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <Substitution>
                        <Player color="Magenta" number="1" state="IN"/>
                        <Player color="Magenta" number="2" state="OUT"/>
                </Substitution>
        </ControlEvent>
</RefereeBoxProtocol>

<!-############################################################################-->
<!-- Options
        color="Cyan"
        color="Magenta"
        color="Yellow"
        color="Blue"
        color="Red"
        color="Green"

        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"

        role="Goalkeeper" | "GK"
        role="FieldPlayer" | "FP"

        state="PlayerOut" | "OUT"
        state="PlayerIn" | "IN"
        state="PlayerSuspended" | "SU"
        state="PlayerSentOff" | "SO"
-->
```

A.13 Timer

```
<!-- Timer message -->
<!-- this message is used to Describe time flow and to set game period and
timer state  -->
<!-- ############################################################################ -->
<!-- Complete/Server/Log version -->
<RefereeBoxProtocol version="2.0">
        <From IP="xxx.xxx.xxx.xxx" ID="0"/>      <!--optional tag to sender-->
        <ControlEvent>
                <Timer period="PreGame" totalPeriodTime="1" totalGameTime="2"
                        playedPeriodTime="3" playedGameTime="4"  running="true"/>
        </ControlEvent>
</RefereeBoxProtocol>


<!-- Sender Control version -->
<RefereeBoxProtocol version="2.0">
        <ControlEvent>
                <Timer period="FirstHalf" running="true"/>
        </ControlEvent>
</RefereeBoxProtocol>

<!--
        totalPeriodTime  => Seconds since period start.
        playedPeriodTime => Seconds whith game running since period start.
        totalGameTime    => Seconds since First Half start.
        playedGameTime   => Seconds whith game running since First Half start.
        running          => true: game clock runnung, false: game clock stoped
-->

<!-- ############################################################################ -->
<!-- Options
        period="PreGame"
        period="FirstHalf"
        period="HalfTime"
        period="SecondHalf"
        period="PenaltyShootOut"
        period="EndGame"

        running="true"
        running="false"
-->
```

Apêndice B

Mensagens Codificadas em Caracteres

Command Description	Cyan Team	Magenta Team
Halt	H	
Stop	S	
Start	s	
End game	e	
Begin first half	1	
Begin half time	h	
Begin second half	2	
Free kick	F	f
Goal kick	G	g
Trow in	T	t
Corner kick	C	c
Penalty kick	P	p
Corner kick	C	c
Kick off	K	k
Goal	A	a
Drop ball	N	

Tabela B.1: Protocolo da *Referee Box* 2005 e 2007.

Apêndice C

Dinâmica dos Objectos do Núcleo

No sentido de cumprir os requisitos enunciados na secção 3.1, mais precisamente os requisitos de segurança da tabela 3.2, todos os módulos periféricos que extraem ou introduzem informação no núcleo passam por um processo de registo.

No diagrama de sequência da figura C.1 são apresentados os traços gerais do fluxo de integração de uma mensagem de registo de uma equipa no sistema. Na tabela C.1 está uma breve descrição desta sequência, onde se apresenta o arranque do sistema e se assume que actor que lança o sistema é um árbitro.

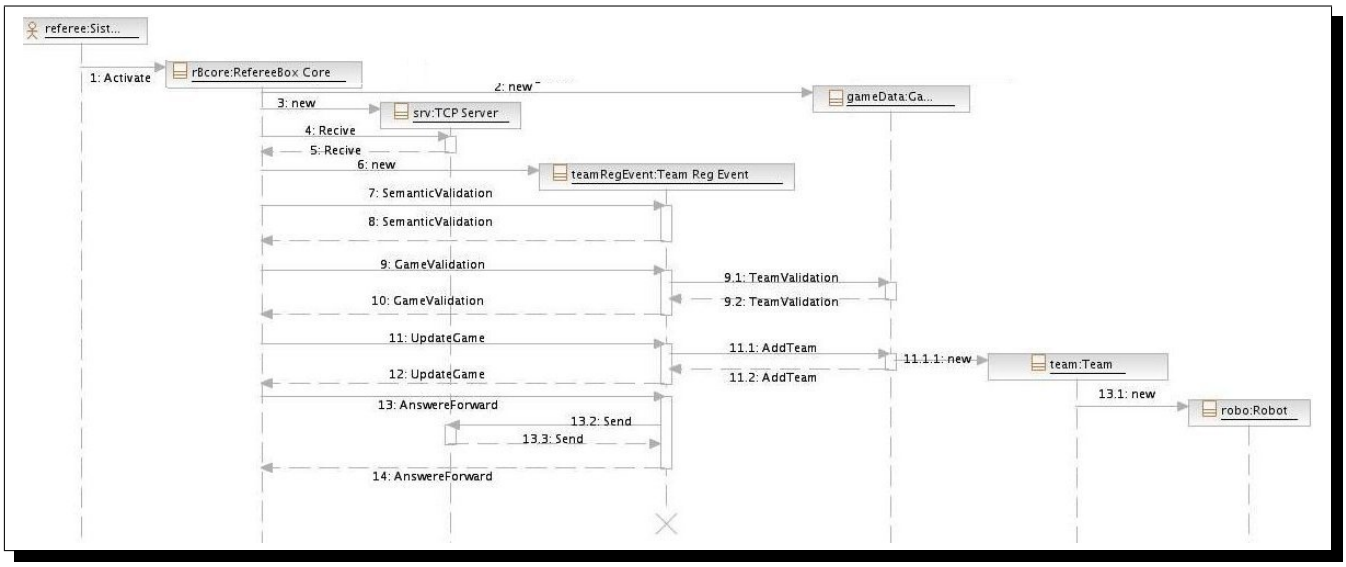


Figura C.1: Diagrama de sequência (*UML sequence diagram*) da integração de uma equipa no núcleo.

A finalidade última do funcionamento do núcleo da *Referee Box* 2008 é integrar eventos de jogo como, por exemplo, faltas e cartões.

No diagrama de sequência da figura C.2 é apresentada a sequência típica da integração de uma falta e na tabela C.2 uma breve descrição desta. Nesta sequência assume-se que o núcleo já foi lançado, ou seja, já existe o objecto que agrega o registo de jogo (*rBCore*) e o que implementa a interface com a rede (*srv*).

1 : Activate	Lançamento do núcleo da *Referee Box* por um actor do sistema.
2, 3 : new	Criação dos objectos de rede e registo de jogo.
4, 5 : Recive	Recepção de uma mensagem da rede.
6 : new	Criação do objecto de evento, mais especificamente um objecto de registo de equipa.
7, 8 : SemanticValidation	Validação da integridade semântica da mensagem de registo.
9, 9.1, 9.2, 10 : GameValidation	Validação do evento no contexto de jogo e do sistema.
11, 11.1, 11.1.1, 13.1, 11,2, 12 : UpgateGame	Integração do evento de registo de equipa no sistema e consequente criação dos objectos com os dados da equipa e robôs.
13, 13.2, 13.3, 14 : AnswerForward	Resposta e/ou reenvio da mensagem integrada.

Tabela C.1: Descrição da sequência de integração de uma equipa no núcleo.

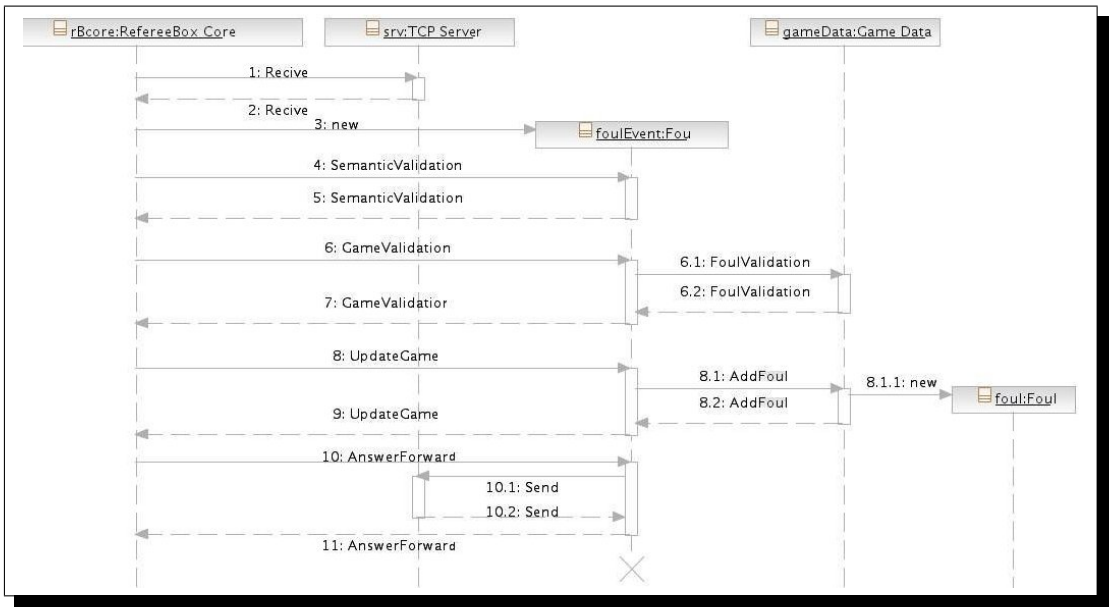


Figura C.2: Diagrama de sequência (*UML sequence diagram*) da integração de uma falta.

No diagrama de sequência da figura C.3 é apresentada a sequência típica da exibição de um cartão a um jogador e na tabela C.3 uma breve descrição desta. Nesta sequência omitiu-se a resposta do sistema e assumiu-se que o núcleo já estava a correr, ou seja, já existe o objecto que agrega o registo de jogo (*rBCore*) e o que implementam a interface com a rede (*srv*), assim como se assume que já existe um objecto equipa (*team*) e um robô (*robot*) com as características descritas no evento (*cardEvent*).

1, 2 : Recive	Recepção de uma mensagem da rede.
6 : new	Criação do objecto de evento, mais especificamente um objecto de falta para uma equipa.
4, 5 : SemanticValidation	Validação da integridade semântica da mensagem de falta.
6, 6.1, 6.2, 7 : GameValidation	Validação do evento no contexto de jogo e do sistema.
8, 8.1, 8.1.1, 8,2, 9 : UpgateGame	Integração do evento no sistema e consequente criação de um objecto com os dados da falta.
10, 10.1, 10.2, 11: AnswerForward	Resposta e/ou reenvio da mensagem integrada.

Tabela C.2: Descrição da de sequência de integração de uma falta.

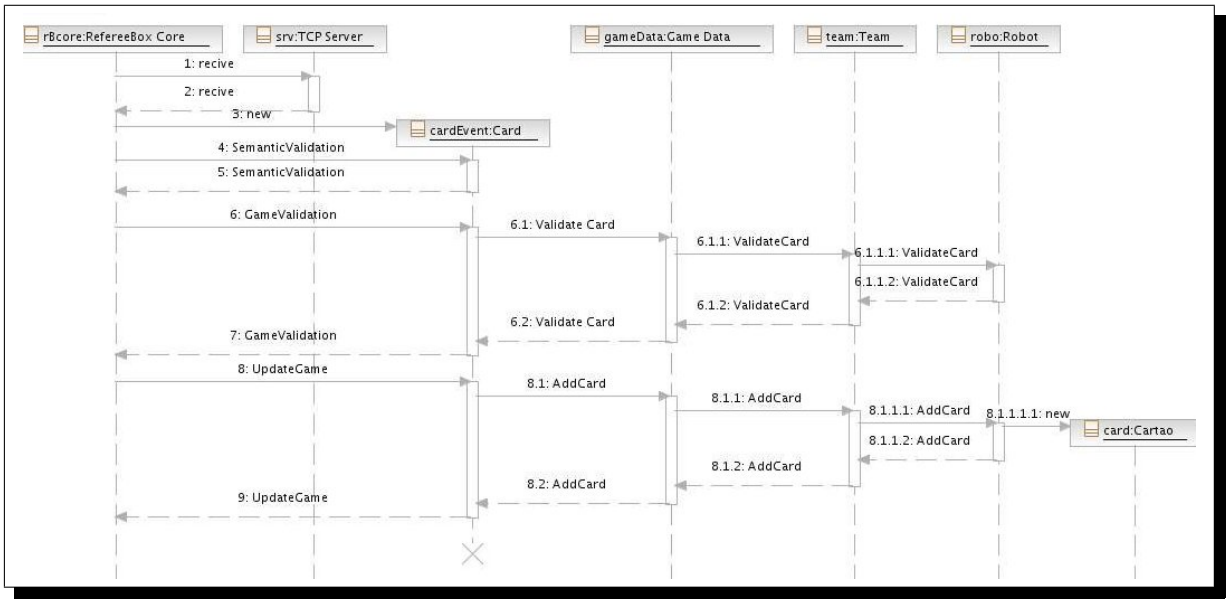


Figura C.3: Diagrama de sequência (*UML sequence diagram*) da integração de um cartão no núcleo.

Não é de mais referir que as sequências apresentadas (fig. C.3, C.2, C.1) não transparecem a total complexidade dos fluxos, principalmente porque muitas das funções despoletam sequências alternativas que não estão modeladas nos diagramas em causa.

1,2 : Recive	Recepção de uma mensagem da rede.
3 : new	Criação do objecto de evento, mais especificamente um objecto de cartão para uma jogador.
4,5 : SemanticValidation	Validação da integridade semântica da mensagem de cartão.
6,6.1, 6.1.1, 6.1.1.1, 6.1.1.2, 6.1.2, 6.2,7 : GameValidation	Validação da do evento no contexto de jogo e do sistema.
8,8.1, 8.1.1, 8.1.1.1, 8.1.1.1.1, 8.1.1.2, 8.1.2, 8,2, 9 : UpgateGame	Integração do evento no sistema e consequente criação de um objecto com os dados do cartão.

Tabela C.3: Descrição da sequência de integração de um cartão no núcleo.

Apêndice D

Referee Box 2008 Report

Game Report RoboCup Robotic Soccer - Middle-Size League

D.1 Sumary

Tournament	robotica2008aveiro
Game Number	0
Date (yy/mm/dd)	7/5/2008
Start Time	17:01:50
End Time	17:50:40
Main Referee	Sertgio
Final Result	CAMBADA - 1 / ISEP - 1

D.2 Teams

Colour	Cyan
Name	CAMBADA
Team Leader	Jose Luis Azevedo
Nr. of Players	6

D.3 Statistics

Match Times	total: **1:12:05** - (first half : **16:02**, half time : **37:03**, second half : **19:00**)
Game Time	25:02
Play-off Time	11:36
Net Time:	13:26

Dropballs	0

	CAMBADA	ISEP
Goals:	3	3
Red Cards	3	3
Yelow Cards:	3	3
Blue Cards	3	3
Free Kicks	3	3
Goal Kicks	3	3
Trow-ins	3	3
Corner Kicks	x	x
Repairs	6	2
Substitutions	1	1
Minimum players nr.	1	1
Average players nr.[1]	1	1
Maximum players nr.	1	1

1

[1]This number represents the the weighted mean of the robot number with the time they stand in the field.

D.4 Registers

D.4.1 Teams

Cyan Team(s)

Team name	Team leader	Team email
ISEP	xpto@isep.pt	Albergaria

IP	ID	Reg. period	Reg. time	Team type
	0	1st-Half	0	DirectTeam

Lineup players		Reserve players	
Number	Role	Number	Role
1	G.K.	7	F.P.
2	F.P.	8	F.P.
3	F.P.	9	G.K.
4	F.P.		
5	F.P.		
6	F.P.		

Magenta Team(s)

Team name	Team leader	Team email
CAMBADA	xpto@ua.pt	JLA

IP	ID	Reg. period	Reg. time	Team type
	0	1st-Half	0	DirectTeam

Lineup players		Reserve players	
Number	Role	Number	Role
1	G.K.	7	F.P.
2	F.P.	8	F.P.
3	F.P.	9	G.K.
4	F.P.		
5	F.P.		
6	F.P.		

D.4.2 Referees

Referee name	**Referee Type**		**Referee email**	
Sertgio	MainReferee		NoEmail	
IP	**ID**	**Reg. period**	**Reg. time**	**Control type**
	0	PreGame	14	GlobalControl

Referee name	**Referee Type**		**Referee email**	
Tafula	AssisReferee		NoEmail	
IP	**ID**	**Reg. period**	**Reg. time**	**Control type**
	0	PreGame	14	GlobalControl

Referee name	**Referee Type**		**Referee email**	
Neves	TimeKeeper		NoEmail	
IP	**ID**	**Reg. period**	**Reg. time**	**Control type**
	0	PreGame	14	GlobalControl

Apêndice E

Referee Box 2005 Report

GAME SHEET

Game Information

FIELD	A [ ] B [ ] C [ ] D [ ]	DATE/TIME	
Team A		Team B	
Team Leader A		Team Leader B	
Team A Color		Team B Color	
Nr. of robots A		Nr. of robots B	
Score A		Score B	

Referees

Main Referee		Assistant Referee	
Blue Goal Assistant		Yellow Goal Assistant	

Infringements

Robot #	Yellow	Out	Yellow	Expulsed	Robot #	Yellow	Out	Yellow	Expulsed
1									
2									
3									
4									
5									
6									
7									
8									

Goals

Robot #	Time	Robot #	Time
1			
2			
3			
4			
5			
6			
7			
8			

Signatures

Main Referee		Assistant Referee	
Blue Goal Assistant		Yellow Goal Assistant	
Team Leader A		Team Leader B	

Figura E.1: Relatório de jogo gerado pela *Referee Box* 2005.